2

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

SECURITY EVALUATION OF UNIX NETWORKS

by

Thomas L. Brown

September 1993

Thesis Advisor: Roger Stemp

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE September 1993	3. REPORT TYPE AND DATES COVERED Master's Thesis, July 1991 - September 1993

4. TITLE AND SUBTITLE Security Evaluation of Unix Networks (U)	5. FUNDING NUMBERS
6. AUTHOR(S) Brown, Thomas Lynn	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Administrative Sciences Department Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/ MONITORING AGENCY NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-500	10. SPONSORING/ MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Unclassified/Unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In recent years, computer networks have significantly increased in both complexity and number, and these networks are attractive targets for attack and intrusion. Unix networks being managed by the government and providing access to unclassified sensitive information are particularly vulnerable to attack.

Ensuring the security of sensitive information will be one of the single most important management issues in computer/information security in the foreseeable future. Unfortunately, the number of automated security tools for Unix, as well as the number of computer security experts within DOD, has not increased sufficiently to keep up with the improvements in technology.

The author proposes the concept of a security toolbox, containing a proposed standard set of automated security tools, to support Unix networks. The toolbox can be used to enhance system security, automating many of the security related tasks required of the network administrator. Additionally, organizational changes will be necessary to improve the availability of computer security advice and assistance. It is recommended that a study of the function and organization of computer security expertise be conducted, so that access to and validation of security tools, along with consistent guidance for network administrators and security officers, can be accomplished effectively. The combination of a security toolbox and expert advice can then help bridge the gap in the development of computer security expertise.

14. SUBJECT TERMS Computer Security; Unix; Information Security; Toolbox	15. NUMBER OF PAGES 127
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

SECURITY EVALUATION OF UNIX NETWORKS

by
Thomas L. Brown
Lieutenant, United States Navy
B.A., University of Texas , 1982

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN
INFORMATION TECHNOLOGY MANAGEMENT**

from the

NAVAL POSTGRADUATE SCHOOL
September 23, 1993

Author: Thomas L Brown
Thomas L. Brown

Approved By: [illegible]
Roger Stemp, Thesis Advisor

Carl R Jones
Carl Jones, Associate Advisor

[illegible]
David R. Whipple, Chairman,
Department of Administrative Sciences

ABSTRACT

In recent years, computer networks have significantly increased in both complexity and number, and these networks are attractive targets for attack and intrusion. Unix networks being managed by the government and providing access to unclassified sensitive information are particularly vulnerable to attack.

Ensuring the security of sensitive information will be one of the single most important management issues in computer/information security in the foreseeable future. Unfortunately, the number of automated security tools for Unix, as well as the number of computer security experts within DOD, has not increased sufficiently to keep up with the improvements in technology.

The author proposes the concept of a <u>security toolbox</u>, containing a proposed standard set of automated security tools, to support Unix networks. The toolbox can be used to enhance system security, automating many of the security related tasks required of the network administrator. Additionally, organizational changes will be necessary to improve the availability of computer security advice and assistance. It is recommended that a study of the function and organization of computer security expertise be conducted, so that access to and validation of security tools, along with consistent guidance for network administrators and security officers, can be accomplished effectively. The combination of a security toolbox and expert advice can then help bridge the gap in the development of computer security expertise.

TABLE OF CONTENTS

TABLE OF CONTENTS

APPENDIX

EXECUTIVE SUMMARY

In this thesis, the author focuses on the importance of improving network security, particularly for government-owned Unix networks. Furthermore, the author proposes the development of a security toolbox and addresses organizational support issues related to the improvement of network security. It should be noted that the security toolbox is a concept which can be implemented quickly, using tools which have already been developed.

The introduction of the thesis sets the stage by reviewing advances in computer technology, the impact of vast computer networks such as the Internet, and the security problems associated with Unix networks. The utilization of Unix networks on the Internet is highlighted as a specific point of vulnerability.

After the introduction, a discussion of recent trends in computer security and some of the key issues for Unix administrators is presented. This section of the thesis will compare some of the better security tools currently available to assist the administrator or security officer in maintaining a Unix network. Each of the tools is presented in such a way that the strengths and weaknesses are highlighted.

Next, the author reviews organizational roles for computer security within the government. The organizations described are the primary repositories of computer security expertise in the government. These organizations include the Computer Emergency Response Team, Computer Incident Advisory Capability, Computer Network Security Response Team, Defense Information Systems Agency, National Computer Security Center, National Institute Standards and Technology, and the National Security Agency; all of which play a role in the advancement of computer security capabilities.

A primary focus of the thesis is addressed in the section on automated security tools for Unix networks and introduction of the security toolbox. The rationale for a security toolbox is discussed, in conjunction with implementation issues. Implementation issues include training, organizational support, tool installation, and tool utilization. Finally, the chapter explores the need for changes in the organizational support and the support required to maintain and update a security toolbox.

The thesis concludes by briefly summarizing the main points. In addition to the summary, the author explores the topic of centralization of computer security expertise and the need for further study and consideration of this issue.

I. INTRODUCTION

Since the inception of the ARPANET in the early 1970's, computer networks have proliferated at a phenomenal pace. From the 1970's until 1987, the number of computer networks connected to the ARPANET steadily increased to approximately 200. Between 1987 and 1989, the number of networks increased to almost 500. [QUAR90: p. 282] In a span of three years, the number of networks connected to the ARPANET more than equalled the number from the previous 17 years.

The ARPANET evolved and matured into what is now referred to as the Internet, which has the potential to interconnect millions of users. [BELS93][QUAR90] This explosion of interconnected computers has significantly increased the availability of information in the research and academic communities, as well as government agencies. [KROL93: pp. 11-13] The current pace of hardware and software improvements will certainly be a driving factor in the increased access to information provided by these interconnected networks, and if the pace of technological advancement continues, in the author's opinion, most homes, offices, government agencies, etc. will be eventually interconnected in a massive array of networks which can transfer information across numerous system configurations.

In a computing environment as described above, network security will be one of the more important management issues in the foreseeable future. As technology allows connectivity between more computers and varieties of networks, the protection of information and resources will become critical. Vast databases of sensitive information will be lucrative and attractive targets for attack, presenting challenges in the management and implementation of information security (INFOSEC).

Unless network security is taken seriously, the competitive advantage needed to sustain U. S. economic development and military capability could be undermined. A global information network, using modern technology, will increase the competition among industrialized countries and will require more sophisticated techniques to protect information. Network security will play a vital role in preserving the security and integrity of sensitive information which must reside in network databases.

In this thesis, the author addresses two aspects of computer security within the government and DOD in particular. The first area of concern deals with minimizing the security vulnerability inherent in the administration of unclassified Unix networks, especially when connected to larger networks like the Internet. The primary method of improving Unix security in the near term is the concept of implementing a security toolbox. The second aspect of computer security which will be addressed is the management role played by the government and possible changes which could be made to improve security management, expertise, and policy.

The more immediate concern is with the administration of unclassified Unix-based networks, managed by the government. Unix security vulnerabilities are continually documented by organizations like the Computer Emergency Response Team (CERT) at Carnegie Mellon University and these vulnerabilities require immediate attention on the part of government computer security personnel. Unix does have powerful security features, but the problem arises from either the inexperience, lack of resources, the addition of programs which undermine security. (See the appendix for an explanation of some critical Unix security issues.) Networks, operated by the government and its contractors, pose numerous vulnerabilities with respect to the protection of sensitive information. Many government-owned Unix networks have gateways to the Internet and those networks are attractive targets for attack and intrusion.

Recent thesis research, conducted by a computer science graduate student from the Naval Postgraduate School, demonstrated that some government networks can be penetrated and exploited with relative ease. [RICH92: pp.20-21] Unclassified government computer systems such as those exploited by Lieutenant Rich are connected to the Internet, and they are vulnerable to attack by hostile governments or criminals. In the author's opinion, the scope of the problem has not yet been systematically addressed. The government remains primarily in a reactive role as demonstrated by the creation of CERT after the Internet Worm infected thousands of host computers, to include many government computers, connected to the Internet. [DCAC]

Automated security tools are a means to increase the security of Unix networks, but these tools should be validated and distributed by government security experts and provided to all managers of government networks. Such tools would assist the administrators or security managers in evaluating their Unix networks, with the ultimate goal of improving the security posture of the network. Unix networks are the primary focus of this paper, but it should be evident that other types of networks may also come under attack, and require improved security management.

There are several tools which have been developed to address Unix network security weaknesses and vulnerabilities, and these tools will be discussed later in this paper. Every network administrator and/or security officer of a Unix system should be familiar with the security tools presented in the following chapters and incorporate those tools in a security toolbox. The toolbox should be protected and restricted to only those personnel involved in network security.

The second aspect of computer security to be addressed in this paper is concerned with improving the computer security advice and assistance provided to managers of government networks. As the Department of Defense (DOD) and other government agencies increase network connectivity to information thoroughfares such as the Internet, it is essential that computer security policy and management procedures be reevaluated. The current trends of technological change have been beneficial in many ways, but the changes often outpace our ability to manage and cope with the new capabilities.

The centralization of computer security expertise (for unclassified systems) should be seriously considered, in light of the significant DOD budget cuts which will be in effect for several years and the difficulty in developing and maintaining computer security expertise. Current management of computer security and INFOSEC is carried out by several organizations, such as the National Security Agency (NSA), the Defense Information Systems Agency (DISA), and the National Institute of Standards and Technology (NIST), along with other supporting organizations and research laboratories. What is needed is centralized management of computer security issues. An organization such as DISA would be a good candidate to provide operational advice, assistance, and policy guidance to all managers of government computer networks. Current practice is for each manager of a computer network to look within their own organization or department for computer security advice and assistance. This method of dealing with computer security must be reevaluated, because of the proven ability of technology to keep ahead of security practice and the difficulty in maintaining computer specialists in government service.

Computer security is proving to be a challenging endeavor as we must attempt to keep pace with the capability of computer technology, which is providing an almost constant improvement in access, transfer, and storage of information. The implications of a lag in security awareness, management, and expertise are not well defined, and indeed many information technology professionals still resist adhering to security guidelines. This thesis will attempt to highlight problems and recommend solutions for Unix networks, as well as propose potential courses of action to improve the management of computer security issues within DOD.

II. BACKGROUND

A. THE ARPANET/INTERNET

1. Scope

The Internet comprises, by some estimates, from 400 to 6000 networks located in 100 countries and connecting from one to six million users.[BELS93: p.13] The scope of the Internet is indeed very large and the impact of such an "electronic village" is significant. The Internet will continue to have far reaching impact in the development of computer security procedures for the use, processing and storage of information.

2. History

The Internet began as the ARPANET in the early 1970's. The Advanced Research Projects Agency (ARPA) provided the funding for an experimental wide area network which would be used to research the impact of temporary outages on computer networks, among other topics. The government was interested in how catastrophic failures or outages could be mitigated in such networks. The initial direction of the research into network protocols was significantly affected by the underlying assumption that the network was assumed to be "unreliable". It is understandable, therefore, why reliability was a major concern in the development of Internet protocols. Advances in packet switching technology, to include the Internet Protocol (IP) were possible because of the network research conducted by ARPA. [KROL93: p.11]

The 1980's brought innovative and substantial changes to the ARPANET, which was considered experimental up through 1983. During this period, the ARPANET architecture had become stable and was also being used for many military operational purposes. The Department of Defense (DOD) wanted to expand network connectivity for the military and decided upon the Internet architecture as a good basis to develop further capability. In 1984, ARPANET was divided into two parts: MILNET was created to provide the military with an unclassified operational computer network, while ARPANET remained primarily a research and development network.

Concurrent with the changes to the military aspects of the ARPANET, the National Science Foundation (NSF) began to expand its network capability by adding five supercomputing centers to the NSFNET, also a part of the ARPANET. The NSF centers were connected via 56kbps leased telephone lines, facilitating the establishment of regional networks. An important factor in the maturation of ARPANET was NSF support of expanded access to the network by educational institutions. NSF promoted the joining of government and research activities with the academic community. [KROL93]

The incorporation of non-IP based networks into the ARPANET was also begun in the 1980's. With the introduction of gateways and routers, more and varied types of networks were able to communicate with the IP based networks. Networks such as BITNET and DECNET could then be connected to the ARPANET, expanding connectivity to other network architectures. [KROL93: pp. 11-13]

The ARPANET was formally disestablished in 1990. Numerous sites which were part of the ARPANET became part of NSFNET, but the MILNET remained under administrative direction of the Defense Information Systems Agency (DISA). [DDN: pp. 9-10] The experimental network that began as the ARPANET evolved into what is now referred to as the Internet.

The Internet is not a single network or organization, but many networks and organizations. The NSFNET and MILNET have comprised the backbone of the Internet because of the high speed links between various regions. The Internet has grown quickly in recent years and new advances in communications technology will have a significant impact on future development. The standards of operation for the Internet are governed by the Internet Architecture Board (IAB), a "...group of invited volunteers...", which meets periodically to decide on network architecture issues.

Coordination of the practical network aspects of the Internet is carried out by the Defense Data Network, Network Information Center (DDN-NIC). [QUAR90: p.279] The Internet continues to expand as organizations and individuals want more access to information. Expansion of the Internet will certainly mean increased international connections in the future, and with expanded access will come the need for more sophisticated security capability.

B. SECURITY IMPLICATIONS OF THE INTERNET

1. Security and Early Internet Development

The ARPANET began as an experimental network in the days when mainframe computers were the primary means of computing. Personal computers had not yet made an impact in the marketplace and computer security was primarily confined to mainframe operation and control procedures. The explosive developments in computer hardware and software in the 1980's significantly increased networking capability, but the new capability also increased vulnerability to illegal or unethical use of computer systems.

Mainframe/mini-computer systems were the primary research computers in the 1970's and were developed with auditing and other control mechanisms which could be used to provide an acceptable degree of security. The nature of mainframe computing and the investment required for the technology ensured that physical security and supervision were considered. Connections to computers outside the organization were the exception rather than the rule. When smaller and more powerful computers were introduced in the 1980's, security functions were not designed into the systems. The technology progressed so quickly that operating system software was standardized before the need for network security was well understood. In the infancy of personal computer (PC) development, the industry did not know the direction that personal computing would take; therefore, security ramifications could not be predicted. Many commercial companies are now developing systems designed with security as a major component, and much of this new development has been initiated and sometimes mandated by the government.

The Department of Defense Trusted Computer System Evaluation Criteria (commonly referred to as the Orange Book) is an example of how the government has attempted to define specific levels of trust for computer systems. The formalization of such security standards and guidelines has been a necessary step for the continued development of computer security. The debate on computer and network security issues will certainly continue as technology continues to provide more capability.

As the computer industry and the Internet have matured, the need for network security has taken on increased importance. The "electronic villages" of computer networks now pose a formidable array of security vulnerabilities for the computer security professional. With more information stored in electronic databases and those databases being set up to be accessed via computer networks, the importance of securing sensitive information becomes apparent. [HOLB91: pp. 5-6]

2. Internet Security

Security vulnerability, with respect to the Internet, has not been adequately addressed if one considers the hardware and software capability which is now available. The first formal attempt to provide security guidance for Internet users and hosts was introduced in Request For Comments (RFC) 1244, published in 1991. RFC's are the means by which the Internet community sets standards and provides information or guidance. The Site Security Handbook (RFC 1244) is an informational RFC and it is meant to be used as a guide for improving security on the Internet. [HOLB91]

RFC 1244 covers a wide variety of topics which impact site security. Topics such as risk assessment, security policy, physical security, audits, incident handling, and recovery procedures are discussed. The document is not intended to provide specific guidance on security but to provide a basis for developing site security policies and procedures.[HOLB91: pp. 3-4] The editors of RFC 1244 acknowledged that the handbook would not be a complete reference; however, in this author's opinion the overall objective of providing a valuable resource to the Internet community was certainly achieved.

It is unlikely that strict security procedures will be implemented on the Internet, due to its scope, size, and complexity. The current trends appear to be emphasizing Local Area Network (LAN) security, along with increasing the security capability of routers and gateways. Implementing security for LAN's and points of connection such as routers and gateways is currently a more realistic goal. Wide Area Network (WAN) and distributed network security is a more difficult problem, due to the scope of such networks and the variety of security problems which can be encountered. Security concerns aside, the Internet will probably be a model for the "electronic highway" being proposed by the political leadership. Secure protocols may eventually be implemented on the Internet, but for the foreseeable future it will primarily be up to the local network constabulary to regulate traffic and maintain security in their particular area.

C. UNIX AND THE INTERNET

1. Unix Overview

The Unix operating system "...was not designed from the start to be secure. It was designed with the necessary characteristics to make security serviceable." [GARF91: p. 8] The statement above indicates the manner in which Unix was developed, as stated by Dennis Ritchie, one of the original developers. Unix was designed for the "open" environment of universities and research institutions and was rewritten in the C programming language during the early 1970's; security was not a critical factor in the development process. The security that was originally included in Unix was intended more to protect the operating system from being "crashed."

Another factor affecting Unix is the fact that it was not developed to operate while connected to outside computer resources such as wide area networks (WAN's) or distributed networks. The newer types of networks did not become widely used until the 1980's. New technologies and the proliferation of high speed communication lines and networks have significantly altered the environment in which Unix systems must operate. The threat to Unix based networks will not be substantially diminished in the near term; therefore, automated security tools and procedures must be used by network administrators. [GARF91]

2. Unix Systems on the Internet

Unix has been a very popular operating system in the academic and research communities because of its flexibility and capability to communicate. Many of the host computers on the Internet use some version of the Unix operating system. [GARF91: p. 8] Unix has powerful communication capability, such as remote virtual terminals, remote file service, electronic mail, electronic directory service, and date/time verification. Many programs have been added through the years to provide more capability for Unix based systems. Because of the original design of Unix, security vulnerabilities are a constant concern when any program is added to the system. (See the appendix for further amplification.) Indeed, programs have caused serious vulnerabilities, which in turn have allowed illegal access to those systems which have not eliminated such vulnerabilities. An example of a program creating a vulnerability is **sendmail**. [FERB93] A bug in an early version **sendmail** program was used by the Internet Worm to send lines of code to a system **debug** command. The extra lines of code created a program which used the system shell to carry out the attack. [FERB93: pp. 247-248]

Newer versions of Unix have been developed in recent years, designed with improved security functionality. Some of the newest versions of Unix have substantial security capability, from a C2 level (Controlled Access Protection) to a B1 level (Labelled Security Protection). [FERB93: pp. 3, 201-203] Unfortunately, many of the Unix systems being used are not the newest versions. Older versions of Unix may not have had software patches added to correct security vulnerabilities, causing many of the Unix systems in use today will remain a potential security problem for years to come, until the older systems are eventually phased out and replaced with more sophisticated versions.

Developers have begun to design secure system software, and future systems will certainly have much more capability in providing access control, operating kernel protection, and general auditing functions. [FERB93] In the interim, tools to improve Unix security have been developed; however, to the author's knowledge there has not been any concentrated effort to develop a standard "toolbox" of software programs or a methodology for utilization of a "toolbox". It is essential that administrators use automated security tools to improve network security and protect sensitive information, and the toolbox concept will fill a needed gap, until newer security programs are developed. [GARF91]

III. NETWORK SECURITY

A. COMPUTER SECURITY TRENDS

1. The Security Threat

Does a threat exist? If the number of security incidents on the MILNET is any indication, then the answer is yes. MILNET security incidents increased from one or two a month prior to 1988 to several incidents each day, in 1989. The primary causes of the security incidents were poor system administration and management. Problems were found in account administration, password administration, and system configuration. [MADR92: pp. 2-3] The proliferation of powerful personal computers and sophisticated software have made the security threat much more real. The means to thwart would be attackers has improved, but advanced technology is available to anyone, and gaining an advantage in security is a zero sum gain.

The information presented in Table 1 provides an indication of the trends in computer attacks and was based on survey information collected in the 1980's. The overall trends are significant and probably indicative of today's trends. The occurrences of attacks increased in each category, with the exception of banks. [CUNN90: p. 69]

Table 1: COMPUTER CRIME SUMMARY

Crganization Type	1986 Attack Percentage	1989 Attack Percentage
Commercial	23%	36%
Banks	18%	12%
Telecommunication Companies	15%	17%
Government	14%	17%
Individuals	Not Reported	12%
Universities	Not Reported	4%

Table 2 shows categories of computer threats and the related percentages as determined by the Datapro Research Corporation report. [CUNN90: p. 68] The source of the threat is not necessarily related to the severity of damage caused by that threat.

Table 2: COMPUTER THREAT

CATEGORY		PERCENT
Physical:	Fire	10%-15%
	Water	10%
	Total	20%-25%
Human (Outsider): Total		1%-3%
Human (Insider):	Errors/Accidents	50%-60%
	Dishonest Employees	10%
	Disgruntled Employees	10%
	Total	70%-80%

It is reasonable to believe that a majority of computer security and integrity problems come from employee error or accident. The problem for the future, however, will be the ever increasing amount of information that resides in computer databases. Those databases will be more and more attractive as targets of unscrupulous persons, willing to gain illegal access to a system, for personal gain. Though most of the security effort may be placed on minimizing errors and accidents, it is important to remember that a compromise of sensitive information could undermine the organization with only one incident.

Table 3 indicates the computer security trends in information technology products from 1985 to 1991. [CUNN90: p. 63] With the exception of anti-virus products, the top four categories of security technology being put to use are all related to access control or protection of information. The trends indicate where the computer security emphasis has been placed. As anti-virus programs and access control mechanisms become more automated and reliable, organizations will tend to seek greater information access via networks. As more network access is required, more emphasis will be placed on network security.

Table 3: SECURITY TRENDS 1985-1991

CATEGORY	PERCENT OF USERS 1985	PERCENT OF USERS 1991	PERCENTAGE INCREASE
Anti-Virus Products	1%	53%	5300%
Advanced Encryption	3%	29%	967%
Smart Cards	5%	36%	720%
Secure Networks	10%	61%	610%
Secure Database Systems	11%	57%	518%
Intrusion Detection Systems	8%	31%	388%
Secure Operating Systems	19%	57%	300%
Audit Analysis Aids	19%	54%	284%
Callback Modems	17%	43%	253%
DES Encryption	19%	47%	247%
Mainframe/Mini Access Control	61%	75%	123%

2. Unix Security

Many system administrators today are in the unenviable position of trying to keep up with normal network administrative duties and network security at the same time. It is not surprising that security has not been given significant attention, except in certain government and specialized commercial activity where security is essential for operation.

In the book, "Unix System Security" by Wood and Kochan, it is recommended that the system administrator be primarily concerned with prevention in four specific areas:

(1) Unauthorized Access,

(2) System Compromise,

(3) Denial of Service, and

(4) Loss of Integrity.

The administrator must protect the system from damage, which can be caused by any of the four categories listed above. [WOOD85: p. 102]

Because Unix was not designed to be secure, the network administrator or security manager has a much greater burden of responsibility. The administrator must be aware of the fact that the Unix systems in use today must be updated on a regular basis to patch security loopholes. Security vulnerabilities are reported by the Computer Emergency Response Team (CERT), once they are found. Maintaining security poses an additional burden for the administrator, particularly in those organizations that do not have a dedicated computer security manager. The following quote from The Hallcrest Report will give some insight into the state of computer security throughout the 1980's and the challenge for IT professionals in the 1990's: "Most security managers are presently ill-equipped, personally and organizationally, to counter the computer security threat, particularly external, electronic intrusion...". [CUNN90: p. 64]

It is not sufficient for an administrator or security manager to learn the Unix operating system and its functions, but it is necessary to be aware of new programs which are added to the system and how they interact with Unix. Any new program added to the system, has the potential of compromising system security, because the program may produce security loopholes unforeseen by the developer.

Since Unix can have multiple users and multiple programs all running at the same time, the potential for security vulnerability and program conflict is significant. Most versions of Unix do have security capability in the form of access controls, where groups and individual users can be granted access to specific portions of the system. The problem arises when security procedures are not followed strictly and improper access is allowed for programs or users. The powerful functions and capabilities provided by the Unix operating system can allow attackers to gain superuser (full system) access if loopholes are not fixed immediately. The potential result of a security loophole can be complete compromise of the system and/or destruction of data or denial of service. [GARF91: pp. 8-9]

B. UNIX SECURITY EVALUATION TOOLS

An important aspect of using security tools is determining how they can best be used in combination. Finding the right combination of tools that use up the least amount of personnel or computer resources is important, especially when these resources are becoming more costly. This facet of tool utilization will be addressed later, in the section on developing a security toolbox.

The security tools mentioned below have been in use for a relatively short period of time, with improvements being added as time and resources were available. Some of the tools have been developed in the past year and a half, with at least one tool (Ice Pick) which has not yet been released for use outside of the Naval Research Lab (NRL). The veteran security tool for Unix is the COPS collection of programs, which has been available on the Internet since around 1989.

There are sources of information available on the Internet which provide a description of each of the tools and what they are designed to do. The intention in this paper is to summarize the capability of each tool, and highlight the differences and/or weaknesses of each tool in comparison to each other. It should be noted that the weaknesses described are meant to focus on areas which should be addressed in the future, as security tool kits become more mature and powerful. The weaknesses do not necessarily refer to inadequacy of the current operation of the programs, unless specifically stated as such.

There is not a "single" source which provides the information presented in this paper. One of the main problems with keeping abreast of Unix security tools is the fact that there is no single repository of security programs and tools. Part of the problem is the dichotomy between commercial and government developed software. Programs such as Ice Pick have been developed in a military research facility and are not initially released to the general public. On the other hand, programs such as COPS [COPS91] were developed in the academic environment and have been freely available to all users on the Internet for some time.

Though not all of the tools discussed here are available to the general public, they should be available to most administrators of government Unix systems. The only exception is Ice Pick, which has not yet been fully developed. The following tool descriptions will provide additional information on government and non-government Unix security programs. By providing a comparative listing of capabilities it should be easier for an administrator to make a more informed decision about which security tools to acquire. It is expected that anyone using such programs has the necessary training and the authority to install the tools on their system.

1. Computer Oracle Password and Security System (COPS)

a. Capability

COPS is a collection of programs which are designed to check various security aspects of a Unix system. The programs were developed by Dan Farmer and other contributors to help system administrators find and eliminate security flaws or vulnerabilities in their systems. Security vulnerabilities are reported to the administrator, but they are not fixed by the COPS program. The administrator must take action to correct security problems found by COPS. The programs are very flexible and can be modified for different systems, but it takes an experienced administrator to be able to get the most benefit from COPS.

Included in COPS is the Kuang program, developed by Robert W. Baldwin. Kuang is an expert system which uses rules to determine if the system can be compromised. The administrator gives the program a goal, such as becoming a super-user, and the program uses its rules and knowledge of the security mechanisms in Unix to find out if super-user access can be acquired. [COPS91] [BALD87]

COPS is currently designed to run on most Berkeley Software Distribution (BSD) and System V versions of Unix. It is also important to note that many of the security tools developed after COPS have used some of the methodology and in some cases the actual programs included in the original COPS distribution. The following list includes some of the more critical checks that are conducted by COPS (The functionality of critical Unix programs and files mentioned below are explained in the appendix.):

(1) Permissions for devices, directories, and files.

(2) Password selection.

(3) Checks of the security, format, and content of password files and group files.

(4) Checks files and programs that are run by **cron** or **/etc/rc*.**

(5) Various facets of root-SUID (see the appendix) files, such as writeability or whether the file is a shell script.

(6) Cyclic Redundancy Checks (CRC) of key files or binaries. The CRC is then used to detect future changes.

(7) Check writeability of start-up files or home directories.

(8) Check the setup of anonymous File Transfer Protocol (FTP).

(9) Checks for hidden shells and other potential problems such as unrestricted **tftp** or SUID uudecode.

(10) Checks various root functions, as well as whether the current directory is in the search path, whether NFS mounts are unrestricted, if a "+" sign is in the **/etc/host.equiv** file, etc.

b. Weaknesses

(1) COPS must be run on a regular basis to detect problems, rather than run as a real-time monitoring system to alert the administrator of problems as they occur. COPS can be set up to run automatically at set intervals, but the programs must be configured so that only changes since the last check are reported.

(2) COPS does not use a Graphical User Interface (GUI). It is a command line driven program in which the user types the appropriate command to begin execution of the programs. The administrator must have properly configured the programs to conduct checks on specific systems. Filters must be set up by the administrator to eliminate redundant reporting. A GUI would make manipulation of the programs simpler and easier for an inexperienced administrator to understand.

(3) COPS does not provide a graphical representation (mapping) of the network, to assist in vulnerability assessment.

(4) No on-line HELP is provided with the COPS package.

2. Security Profile Inspector (SPI)

The SPI security tool for Unix was developed under the sponsorship and technical guidance of several organizations, which included the U.S. Department of Energy, the U.S. Air Force Cryptologic Support Center, and the Defense Intelligence Agency. SPI/Unix version 2.1 was released in July 1992, and is currently maintained by the Computer Security Research Center at the Lawrence Livermore National Laboratory. The main goal of SPI is to assist administrators and security managers in maintaining the security and integrity of the system. The SPI software performs most of the same functions as COPS package with some additional functionality, but it was developed with a different user interface. The primary capabilities and interface modes are listed below:

a. Capability

As an integrated security package, SPI is more functional than COPS, but this capability requires more hardware resources, such as increased disk space. SPI substantially incorporates the functions of COPS (including the Kuang program), along with an integrity checker and an on-line HELP facility. The following capabilities are provided by SPI/Unix:

(1) SPI/Unix uses a menu type user interface. There are six distinct parts to the security package, which include the Quick System Profile (QSP), the Access Control Test (ACT), the Password Security Inspector (PSI), the Binary Inspector Tool (BIT), the File Inode Change Detector (FCD), and the File Data Change Detector (DCD). The QSP, ACT, PSI, and BIT portions of the package test for vulnerabilities in the Unix system. The FCD and DCD portions of the package are used to test for possible intrusion into the system.

(2) SPI/Unix QSP conducts a set of security checks based on and adapted from the checks developed for COPS.

(3) SPI/Unix ACT is designed to check for dependencies in access control files used by Unix. This portion of SPI was adapted from the "Kuang" tool, developed by Bob

Baldwin. ACT is rule-based and seeks to achieve a goal, such as finding dependencies that lead to a compromise of root access of the system. Other goals can be requested, such as gaining write access to a specific file, executing commands as a particular user, or executing commands as a particular group member.

(4) SPI/Unix PSI is used to check the Unix passwords that do not provide adequate protection. PSI uses word dictionaries and word permutations to determine if the password can be easily acquired. Three dictionaries are used to check passwords, which include commonly used English words, local idiomatic words, and a listing of trivial words. The local dictionary can be modified to include words specified by the administrator. PSI will also check for old passwords, which could indicate dormant accounts, or accounts without passwords.

(5) SPI/Unix BIT checks the system binaries to determine if the most recent software patches have been installed. The digital signature (RSA MD5) associated with each critical binary file on the system is computed by BIT. The digital signature is then compared to a database of signatures which is included in SPI/Unix. The database consists of digital signatures of critical binary files, to include versions that are known to be out of date and those that are current. Once the comparison is completed, the program informs the administrator of the results and makes further recommendations, if updates are required. *It is important to remember that BIT can only check for those binaries distributed prior to the current version of SPI/Unix.*

(6) SPI/Unix FCD checks the inode attributes of system-critical files to determine if the files have been modified. The attributes checked include checksum, size, modification time, permissions, number of links, etc. The attributes are checked against a database originally generated by the administrator by using the "snapshot" function provided in SPI. The snapshot is acquired during the installation of the SPI package. The snapshot can also be taken by using the Unix *istat* program.

(7) SPI/Unix DCD creates a unique signature for each critical file on the Unix system. The signature is created by the MD5 Message-Digest Algorithm developed by RSA Data Security, Inc. A database of the original signatures is acquired during the installation process. A snapshot of the signatures can also be obtained by using the Unix *sstat* program.

(8) From the main menu, the administrator can choose which portion of the SPI program to use, as well as view output reports, schedule run times, manage parameters of the system, and check the status of an active process.

(9) The security portions of SPI are run in the background, once chosen from the main menu. This allows the user to make other selections or conduct other functions simultaneously.

(10) On-line HELP is provided by the SPI/Unix package.

(11) Like COPS, SPI can be ported to various versions of the Unix operating system. SPI has been tested on BSD, System V, and Sun OS.

b. Weaknesses

(1) SPI/Unix must be set up to run on a regular basis, rather than run as a real-time monitoring system to alert the administrator of problems as they occur. SPI can be configured to run automatically at set intervals.

(2) SPI does not use a Graphical User Interface (GUI). It is a menu driven program in which the user selects which portion of the program to execute. A GUI would make manipulation of the programs simpler and easier for the novice to understand.

(3) SPI does not provide a graphical representation of the network, to assist in vulnerability assessment.

3. Ice Pick

Ice Pick is a security package being developed by Jeff Humphrey, of Kaman Sciences Corporation, under contract to the Naval Research Laboratory. Ice Pick was written to provide a security tool which can be used to find security vulnerabilities and attempt to compromise system security. The package consists of a testing daemon (program) and an interface program. Ice Pick can be used to test the security of a system by attacking portions of the system and attempting to find security flaws.

The testing daemon will try to compromise security on the target system, and then report the results to the interface program which can provide a report of the test results. Ice Pick should be used in conjunction with other security tools. For example, COPS or SPI can be used to find potential vulnerabilities. After appropriate corrective actions have been taken, Ice Pick can be used to verify that the corrective actions were effective. Ice Pick cannot prove that a system is secure, but it can help in assessing system vulnerabilities and provide a measure of confidence in the system's overall security posture.

The current version of Ice Pick is primarily designed to test for known operating system vulnerabilities. The package is being developed with the assumption that additional testing modules will be added to the package, to increase the overall effectiveness of the tool.

a. Capability

(1) The program tests the security and configuration of the following system functions: ftp, E-mail, tftp, export of file systems, X-windows, system command interfaces, remote network connections, privileged accounts, communications sockets and links, networking applications, and security monitoring stations.

(2) The program keeps detailed logs of its activity, as well as logs of past testing information.

(3) Testing can be conducted at varied levels, from level 1 which checks for known system vulnerabilities, to level 7 which attempts to compromise the system.

(4) The package uses X Windows, providing a Graphical User Interface to manipulate the program.

(5) A visual representation of network nodes can be displayed on the screen, with color coding for system status.

b. Weaknesses

(1) Ice Pick is not yet fully developed and has not been tested by computer security professionals outside of NRL.

(2) Full documentation and code is not yet available.

4. Tripwire

Tripwire is a security tool developed at the Purdue Research Foundation of Purdue University, by Gene Kim and Gene Spafford. The primary purpose of the software is to verify the integrity of critical system files residing on a variety of Unix systems. The integrity of a file is checked by creating a digital signature of the file and comparing the signature with a stored version of the signature. A system administrator who has properly set up Tripwire can use the security tool to ensure that an attacker has not changed any critical files on the system.

Several types of digital signatures can be created by Tripwire. The signature algorithms used in Tripwire include MD5, Snefru, MD4, MD2, CRC-32, and CRC-16. The first four algorithms are cryptographic algorithms, also referred to as message digests or one-way hash functions. The cryptographic technique is used to ensure that any change to the original file creates a vastly different signature. By using a 128 bit signature, these algorithms are quite difficult, if not impossible to break in any reasonable length of time. The last two algorithms, CRC-32 and CRC-16, generate checksums using polynomial division. This method is significantly less secure than the cryptographic algorithms, but much faster.

The developers of Tripwire recommend that MD5 and Snefru both be used to create a digital signature of each critical file. Both signatures should be used in combination to verify file integrity. By using two signatures, it would be almost inconceivable that a file could be altered in any way and not change the signature.

a. Capability

(1) Tripwire can be configured to run automatically at set intervals, but it must be run on a regular basis to be effective.

(2) The algorithms used by Tripwire give a very strong assurance that file integrity is maintained. If properly configured, it would be virtually impossible for an attacker to alter files on the system, without being detected.

(3) Tripwire is able to send results of its integrity checks to the system administrator via mail.

(4) Tripwire is very flexible and can be setup to create various combinations of signatures, as required. The capability also comes at the price of requiring a relatively skilled administrator.

b. Weaknesses

(1) Tripwire must be configured to run on a regular basis, rather than run as a real-time monitoring system to alert the administrator of problems as they occur.

(2) Tripwire does not use a Graphical User Interface (GUI). It is a command-line driven program. A GUI would make manipulation of the program simpler and easier for an inexperienced administrator to understand.

(3) An experienced system administrator is needed to understand which files should be checked and at what regularity. Though Tripwire is very flexible, it must be configured properly in order to provide a sufficient level of protection.

(4) Execution time of the cryptographic algorithms can be quite lengthy, using valuable CPU time.

5. TCP Wrapper

TCP Wrapper is a program developed by Wietse Venema of Eindhoven University of Technology in the Netherlands. The program is designed to intercept requests for Internet services such as telnet, finger, ftp, exec, rsh, rlogin, tftp, talk, comsat, and other services. Once the request is intercepted, TCP Wrapper logs in pertinent information such as the name of the remote host, the name of the service which was requested, and a time stamp. After recording the information, the program can verify the validity of the request before passing it on to the appropriate server.

a. Capability

(1) TCP Wrapper uses the Unix syslog facility to log information.

(2) The log information can be used to document activity should an attacker attempt to compromise the system.

(3) The program has some capability to detect if a host computer is trying to use another hosts name to spoof the system into providing services. If spoofing is detected, TCP Wrapper can drop the connection to the suspect host.

b. Weaknesses

(1) The administrator must tailor the information provided by TCP Wrapper so that the sheer volume of information is not so overwhelming that it obscures suspicious activity.

(2) In a large network, many servers may have to provide service to other Internet hosts. In this case, TCP Wrapper must be placed on any server which is required to respond to Internet service requests. The administrator must configure the network so that TCP Wrapper is a value-added addition and not an inefficient use of computer resources. Minimizing the number of servers can help in this regard.

6. Crack

Crack is a program written by Alec D. E. Muffett, a Unix Software Engineer in Aberystwyth, Wales, United Kingdom. It is currently in its fourth version and has been used extensively by system administrators and others to crack the encrypted passwords stored in the Unix system. The program takes the Unix password file and produces a sorted list. The list from the password file is scanned for useful information and that information is merged with dictionaries of words supplied in the program. The merged data is then used to generate passwords to compare with the passwords residing in the password file. Several passes are made over the password file ... on rules specified in the Crack program, and any match to a password is saved and reported. [MUFF92]

Crack is a useful program to check for easily guessed passwords, but it would be much more preferable to have a program to check, in real time, each password as it is assigned to determine if it meets secure criteria. There have been some recent programs written to assist administrators in the task of assigning secure passwords, by ensuring easily guessed passwords are not chosen by a user. One such program is **Passwd+**, written by Matt Bishop. Programs such as **Passwd+** can check for an appropriate password length, a proper combination of characters and numbers, or compare the password against a dictionary of easily guessed passwords. Once users understand the importance of choosing and using secure passwords, utilization of the programs like Crack and **Passwd+** could become an extinct practice.

a. Capability

(1) Crack has the ability to spread the workload of cracking passwords over several machines on a network. This ability can significantly reduce the amount of time needed to try the various dictionaries and rules.

(2) Dictionaries can be added to Crack as necessary.

(3) The program is self-installing.

(4) The program can send a warning to a user whose password has been cracked, as well as to the administrator.

b. Weaknesses

(1) Crack must be run on a periodic basis to be effective. A capability to run continuously in the background, checking newly assigned passwords, would be beneficial.

(2) Crack does not use a Graphical User Interface (GUI). It is a command-line driven program. A GUI would make manipulation of the program simpler and easier for an inexperienced administrator to understand.

(3) An experienced system administrator is needed to understand how best to use Crack to improve overall password security on the system.

(4) Utilization of Crack can consume a significant amount of system resources, with the execution time being directly related to the configuration specified by the administrator. Judicious and effective use of Crack is dependent on the skill of the administrator.

IV. ORGANIZATIONAL ROLES

A. TRAINING

Perhaps one of the most important problems facing the information systems manager is the general lack of training and expertise in computer security. Network administrators are often required to learn their duties on the job, with little formal training in computer security. Information reported by the National Center for Computer Crime Data (NCCCD) indicated that in 1989 over 50% of personnel working in the computer field either had not received computer security training or they had learned about security on the job. The NCCCD information was derived from surveys of members of the Data Processing Management Association (DPMA). [CUNN90: p. 76]

One of the primary reasons for lack of support in computer security is a general lack of expertise. The lack of security expertise can in part be attributed to the rapidity of changes in technology, but as previously explained, the government has not been able to provide incentives to attract enough computer science/security specialists. [CONG89] Computer security curriculums in U. S. academic institutions are still in their infancy. Many of the computer security professionals today have gained their expertise through on-the-job experience and not through extensive formal education. The shortfall of computer security education is simply another manifestation of the general lack of computer science training in DOD, which was recognized as a problem in 1983. The following statements describe the state of computer science training, and should sound very familiar; in fact, a 1989 congressional report cited continued problems in computer science [CONG89]:

...The Department of Defense, and certainly the entire Federal government is inundated with computers. As both technology and computer system uses and techniques expanded, the training and education programs stagnated: in fact, many of the programs were reduced in length to save money. Two problems continue to plague the computer science field: neither training nor the specialists have kept pace with the technology evolution. [HEDG83: p. 26]

Undergraduate computer science education in the United States has problems -- serious ones -- which must be fixed... [C]omputer science research is already underfunded, both in absolute terms and in comparison to other scientific disciplines. [CONG89: p. 23]

The government does not yet offer a viable career path for computer specialists (to include security professionals) nor adequate salaries as compared to the private sector. [CONG89] Based on the difficulty of attracting computer professionals to government service, it is essential that the government, with the leadership of DOD, make maximum use of its computer security expertise and obtain the necessary amount of expertise. A study of the proper allocation of computer security functions, between centralized and decentralized operations, could provide a comprehensive and flexible program. But where should the expertise reside? In an era of downsizing in the military, it is difficult to create new organizations, but it is critical that scarce resources be used in as efficient a manner as possible.

From an organizational perspective, it is essential that the top level management be convinced of the necessity to provide resources, in personnel, funding, maintenance, and improvement of network security. Without organizational support for computer security, it will be difficult to maintain security of information, accessible via computer networks. In recent years, there have been significant changes in how the government manages computer security issues. Some changes have evolved from the original charter of organizations like NSA (as modified by NSDD 145), but other changes have occurred because of catastrophic events such as the Internet Worm, which affected over 6,000 host computers on the MILNET and Internet. [DCAC: p. 2-1]

The nature of military and government operations requires that sensitive unclassified information be stored and processed, and such information will need greater protection in the future. Sensitive unclassified information now resides in computer databases and those databases have not been given the amount of attention and resources that have been provided for the protection of classified information. Proper handling of sensitive unclassified information stored on computer networks will require a significant amount of training in computer security procedures.

The support organizations mentioned below have taken on additional responsibilities as the need for computer security has increased. Computer security issues are being addressed, but it is necessary to think about ways to improve on the disparate organizational structures and security standards in general.

B. SUPPORT ORGANIZATIONS

Today, there are a number of government organizations which are tasked to provide computer security assistance. The quality of the services they provide and the consistency of the information promulgated by these organizations is greatly impacted by funding and staffing constraints, as well as the degree of coordination with other computer security professionals. There is a great deal of computer security expertise scattered throughout these organizations, but in the author's opinion, there is a duplication of effort which stems from decentralized management of scarce computer security resources.

Based on the amount of resources required to maintain security expertise in information technology, it would seem that the computer security resources could be better utilized in a centralized structure. Centralization would require a great deal of inter-agency cooperation, but the payoff could be substantial. By using a single organization as the focal point, managers of computer networks would be able to get the benefit of an extensive corporate knowledge base, coupled with tools and advisory resources to improve security.

To highlight the disparate government authorities involved in computer/ information security today, a brief explanation of several organizations is provided below. The organizations are all involved in some aspect of computer security and provide an indication of where computer security resources reside in the government.

1. Computer Emergency Response Team/Coordination Center (CERT/CC)

The Computer Emergency Response Team/Coordination Center (CERT/CC) is located at Carnegie Mellon University. The CERT/CC was created by funding from DARPA in December 1988. The impetus for creating the organization was the Internet Worm, which was unleashed by Robert Morris Jr. in November 1988. The worm effectively brought down thousands of computers. Many computers crashed because of the worm, and others simply stopped functioning properly. Some administrators disconnected their computers from the Internet, because of the paucity of information concerning protective measures in the early hours after the worm's release. [KEHO92]

The CERT/CC was formed to fill a hole in the dissemination of information and monitoring of security related incidents. It is a civilian organization with no regulatory powers and its major function is to act as a central point of contact for government and civilian organizations to report security incidents, primarily on the Internet. The CERT/CC also promulgates information concerning security holes found in operating systems and other security related issues. One of the additional services that the CERT/CC provides is the coordination of the CERT TOOLS mailing list. The mailing list is used to promulgate information about security evaluation tools being developed or currently in use by administrators and security practitioners. [GARF91] [HOLB91]

2. Computer Incident Advisory Capability (CIAC)

The Department of Energy (DOE) Computer Incident Advisory Capability (CIAC) is located at Lawrence Livermore National Laboratory. The CIAC consists of four computer scientists, who must provide assistance for all types of computer security incidents. The CIAC is expected to provide assistance on a 24-hour basis, if needed. It must also coordinate security incidents with the emergency response teams of other organizations.

Many of the duties performed by the CIAC mirror those types of duties performed by the CERT/CC; however, the CIAC is primarily responsible for providing computer security expertise for Department of Energy sites. CIAC is also responsible for providing technical assistance to DOE subordinate organizations. The charter of the CIAC encompasses giving advice and assistance on all aspects of computer security as it relates to the DOE. Such a charter is a large responsibility for an organization the size of CIAC and underscores the necessity for tight coordination with other agencies. [GARF91] [HOLB91]

3. Computer Network Security Response Team (CNSRT)

The NASA Ames Research Center Computer Network Security Response Team (CNSRT) is similar to the CERT/CC, but it is responsible for providing a computer security focal point within NASA Ames. Created in 1989, the CNSRT coordinates its activity with the response teams of other organizations. It has provided assistance to other federal agencies, but its primary responsibility continues to be for computer security issues within NASA Ames. [HOLB91]

4. Defense Information Systems Agency (DISA)

The Defense Information Systems Agency (DISA), originally the Defense Communications Agency (DCA), was renamed in 1991. The organization is now in the midst of reorganization due to the downsizing of the military and consolidation of various aspects of information management in DOD. DISA has numerous responsibilities, of which management of DDN security and operations is but one small portion. There is currently an organization within DISA, the Defense Information Systems Security Program (DISSP), that is also in the process of defining its role in information systems security (INFOSEC) in DOD. The downsizing of DOD will cause some amount of instability in organizations such as DISA as they attempt to reorganize for more efficient utilization of assets. DISA is not yet the major contributor in computer security research and expertise, except through its management of DDN. DISA will be required to coordinate with the National Security Agency (NSA) to clarify it role in INFOSEC. NSDD 145 gives the INFOSEC responsibility to NSA, but a key question will be how an organization as large as DISA should fit into INFOSEC management once it is reorganized and consolidated.

5. National Computer Security Center (NCSC)

The NCSC is located near Linthicum, Md. and is an organization of the National Security Agency (NSA). It was established in 1981, as the DOD Computer Security Evaluation Center with a name change to the NCSC in 1985. One of the primary goals of the NCSC is to encourage and provide assistance in making trusted Automated Information Systems (AIS) available for organizations processing classified information. The NCSC has played a key role in developing standards of trust for computer systems. This role is closely aligned to that of NIST which deals with standards for unclassified systems. Through the years there has been much discussion on how the two organizations should coordinate their activity on developing standards in computer security. [DOD85]

6. National Institute Of Standards And Technology (NIST)

The National Institute of Standards and Technology (NIST) Computer Security Resource and Response Center (CSRC) is located in Gaithersburg, Md. The primary mission of NIST is to develop security standards for organizations outside of DOD. NIST was involved with the creation of the CERT/CC in 1988, and has continued their involvement in the development of emergency response capability. NIST has its own 24-hour capability which is provided by the CSRC. Its goal is to provide a point of contact for computer security related information. It operates a computer bulletin board system (BBS) to promulgate computer security information.

NIST also is involved with developing methodologies for computer security evaluation. These tasks are duplicated to some extent by the National Computer Security Center (NCSC), a subordinate organization of the National Security Agency. NIST has provided publications for access via the Internet, and organizations with Internet access can download several publications which deal with computer viruses or computer security. [GARF91] [HOLB91] [CONG91]

7. National Security Agency (NSA)

NSA is primarily responsible for Signals Intelligence activity of the government, for the purpose of collecting and processing national foreign intelligence; however, another important aspect of the NSA charter is its designation as the National Manager for Telecommunications and Automated Information Systems Security, which was stipulated in National Security Decision Directive (NSDD) 145 of 1984. In order to carry out its function under NSDD 145, NSA reorganized its communications security (COMSEC) and computer security (COMPUSEC) divisions into a consolidated information security (INFOSEC) division. NSA's responsibility was also broadened to include unclassified and national security sensitive information, where previously it had been concerned with classified information almost exclusively. With NSDD 145, NSA assumed duties as the government focal point for cryptography, telecommunications systems security, and automated information systems security. [NSDD84]

C. THE FUTURE

In the author's opinion, DOD must be willing to assume a flexible and innovative leadership role, so that establishing a central focal point for operational level computer security issues can become a reality. Changes in communication culture will require significant investment in computer security, primarily because the technology has progressed so fast that the implications of such change are difficult to predict. People understand the need for police and other forms of law enforcement, but the protection of the value and privacy of vast databases of unclassified sensitive information is an intangible need which is difficult to articulate. Without adequate protection, however, information critical to the economy and technological development will be exploited by those governments wishing to gain a competitive advantage based on the capital investment of the U.S.

Establishing a central authority for computer security support will allow the military services to coordinate the use of computer security techniques and tools. Centralization is not a new idea, but it has been consistently resisted by the military services. The Defense Management Report Decision (DMRD) 918, signed in 1992, directed the consolidation of IT under the auspices of DISA. The pace of the consolidation has slowed in 1993 and the military services have objected to how the consolidation is being executed. Part of the consolidation will include INFOSEC, but this particular aspect of the reorganization is in a evolutionary stage, with NSA remaining as the primary authority. [NSDD84]

It is particularly essential that the military services recognize the need for a central focal point for computer security issues. The services may need to reassign personnel to create such an organization, but if organized properly, the long term benefit should surpass the loss in personnel resources. The ability of each of the military services to keep up with technological innovations is impaired because they cannot dedicate enough resources to maintain expertise.

Without wise management of security today, the losses sustained by not adequately controlling sensitive information could potentially undermine national security and/or technological competitiveness. The outcome of competitive economics or future military conflicts will depend on protecting the technological frontiers and managing information wisely. Computer security will be critical to this goal.

V. THE SECURITY TOOLBOX

A. SOFTWARE TOOLS

Unix accommodates networking as an integral part of the system capability. Unfortunately, it takes an experienced Unix system administrator to deal with potential security vulnerabilities associated with Unix. The operating system is very large and complex, and since many system administrators are assigned to the duties with little or no formal training, it is essential that security tools be made available to assist the administrator. [CUNN90]

Several security tools were presented in the last chapter, including Ice Pick, a new tool being developed by the Naval Research Laboratory. These tools were developed because of a growing perception that effectively managing Unix security in a network environment requires the use of automated methods to keep track of system security. Automated security checks will be essential as system complexity increases and as increasing amounts of sensitive and valuable information is stored in computers. When those computers operate in a network, the potential for an attack increases.

Technology continues to improve our capability to manage and store information, but our methods of securing the information has lagged behind. The skill required to protect information on computer networks is much greater than for the traditional security concerns such as physical security. Before the advent of computer networks, physical protection of resources was the most prominent requirement. The new threat today is the accessibility of vast databases of information which can be exploited remotely, by an attacker. This type of vulnerability is certainly not ignored by hostile governments, as demonstrated in recent years by the arrests of computer crackers in Europe, caught illegally accessing U. S. government computer systems.

The need for a dedicated organization focusing specifically on computer security issues was underscored by the infamous Internet Worm incident. The Internet Worm was released by Robert T. Morris on November 2, 1988 and subsequently, the Computer

Emergency Response Team/Coordination Center (CERT/CC) at Carnegie Mellon University was established in the same year. The threat demonstrated by the Internet Worm was the catalyst for creating the CERT/CC. [FERB93: p. 241] [HOLB91] [SPAF88] The CERT/CC began operating in December 1988 with funding from the Defense Research Projects Agency (DARPA). The primary goal of the CERT/CC has been to serve as a principal point of contact for computer security incident reporting. [HOLB91: pp. 43-44] The effectiveness of an organization such as CERT/CC demonstrates how good coordination can be achieved through centralization of computer security expertise.

As the CERT/CC has matured, it recognized the need for automated security tools and started the CERT TOOLS mailing list in 1992. The mailing list serves to facilitate the exchange of information on security tools. The CERT/CC does not have the resources to evaluate and/or endorse particular security tools, therefore, it is up to individual system administrators to select, install, and evaluate these tools. This assumes, of course, that these administrators possess the requisite skills necessary to assess the tool's effectiveness.

What is needed by each administrator of a Unix network is a Security Toolbox which has been evaluated by security professionals. Until such formal procedures are established to address security tool evaluation, the administrator's toolbox should include, at a minimum, the programs discussed in this paper. Other programs should be added as well, such as a program to check passwords as they are assigned to the user.

One of the major problems with maintaining a security toolbox will be the fact that there is no recognized central authority which has evaluated and tested the effectiveness of such programs. This aspect of managing computer security will be addressed in the next chapter.

B. TOOLBOX IMPLEMENTATION

1. Training

A system administrator who has a thorough understanding of his or her specific system is essential to developing and maintaining a secure network. Formal training specific to an organization's configuration and experience on similar systems should be a primary criteria prior to hiring or selecting the system administrator. In addition, formal systems security training is desirable. If formal training is unavailable, it is essential that the administrator attend any computer security conferences, seminars, or workshops that are offered. Conferences are held in various locations throughout the U.S. and are frequently announced in security related articles posted on Internet newsgroups.

An administrator can and should keep abreast of computer security related issues by subscribing to appropriate Internet newsgroups and mailings. A listing of several security related newsgroups available on the Internet follows:

(1) alt.security
(2) alt.security.keydist
(3) alt.security.pgp
(4) alt.security.ripem
(5) comp.risks
(6) comp.security.misc
(7) comp.virus
(8) news.announce.conferences
(9) news.announce.newgroups

Mailing lists are often announced on the newsgroups, along with procedures to get added to the list. An example of a mailing list would be the CERT TOOLS list mentioned earlier. This list is controlled by CERT, and it is restricted to administrators and other personnel with a legitimate interest in computer security.

2. Organizational Support

There are several organizations which provide excellent support in the area of computer security. The following organizations are some of the principal organizations involved with computer security issues: Computer Emergency Response Team/ Coordination Center (CERT/CC) at Carnegie Mellon University; Defense Information Systems Agency (DISA), Department of Energy Computer Incident Advisory Capability (CIAC) at Lawrence Livermore National Laboratory; NASA Ames Computer Network Security Response Team (CNSRT); National Institute of Standards and Technology (NIST) Computer Security Resource and Response Center in Gaithersburg, Maryland.; and the National Security Agency (NSA) in Ft. Meade, Maryland. [HOLB91: pp. 43-46][NSDD84]

3. Security Tool Installation

Once an administrator has acquired security training and knows who to contact with security questions, the next step is to obtain the necessary security tools and install them on their system. The administrator should determine which organizations retain the appropriate security tools and use the "anonymous ftp" capability to transfer the tools to their system. Those tools that cannot be obtained via the Internet should be requested via mail or other means.

After acquiring the appropriate tools, the administrator will most probably need to compile them on the system where they will be used. This process can require some expertise in changing default parameters before compilation. Since there are currently no comprehensive tools which can perform all necessary security functions, a skilled administrator is required for installation and proper configuration of the tools. Care must be taken to install the tools properly, so that the process of installation does not inadvertently create additional security vulnerabilities. For instance, accidentally installing a tool in an unprotected portion of the system can produce a serious security breach of the system and defeat the very purpose for which the tool was designed.

4. Security Tool Utilization

a. Conducting a Baseline Snapshot

The first and most critical action that a system administrator can take is to ensure that original copies of the system software remain un-modified. Next, the administrator must verify that the installed system software has not been modified. This may necessitate reinstallation of the system software from the original write-protected disks or tapes.

Once a "clean" system has been confirmed, a snapshot of all critical files should be stored in a secure location which is inaccessible to unauthorized personnel. To make a snapshot, the administrator can use either the SPI/Unix security package or the Tripwire program. The SPI/Unix package uses MD5 to create a digital signature. If more security is required, then Tripwire should be used, since it can use more than one type of signature in combination, to verify critical files.

Taking a snapshot of system files, in conjunction with good backup procedures, is well worth the investment in time and money. It only takes one compromise of system security to crash a system, but good security practices will ensure the system is back up in a timely fashion.

b. Initial Security Checking

Once the operating system has been installed, verified, and a snapshot taken, the next step is to check various aspects of the system for known vulnerabilities. Either SPI/Unix or COPS should be used for this procedure. If there is adequate memory and computing power available, SPI/Unix is the preferred choice. SPI offers a comprehensive security package in a single tool. Furthermore, since it is a menu driven tool, it is a bit more user friendly. SPI/Unix does require significantly more memory and disk space which may pose a problem on smaller systems. On smaller systems, COPS would be a more logical choice.

Once the tool has been installed and the system has been checked, the administrator should decide what types of checks should be made and the periodicity required in order to best maintain a secure system. The regularity of checks will most likely be affected by system configuration, storage capability, sensitivity of operations, and other factors which must be determined by the individual administrator. Consideration must also be given to how the security information will be archived, when it is collected. Once archived, physical security of the archives is a significant issue. If not secured properly, the archive could be stolen by unauthorized personnel and used to compromise the system.

c. Password Checking

The policy of password selection and assignment could arguably be the most important issue for a system administrator. Good password management can thwart many potential security problems. Good password selection can virtually eliminate the threat of anyone guessing a user's password. It is essential that system administrators instruct their users on the proper selection of passwords, the importance of safeguarding the passwords, and user responsibilities.

The key point is that an administrator must strive to maintain a policy which requires minimum standards of password selection and use. Recommended methods of password selection have been well documented; but there have been password programs developed to check passwords. Programs such as **Passwd+**, developed by Dr. Matt Bishop, can replace the **/bin/passwd** program on a Unix system. The program is designed to discourage selection of a weak password. [REIN93] Without an automated program to check passwords, it is up to the administrator to maintain a strong policy. Password policy information can be obtained from the Department of Defense Password Management Guideline. The guideline recommends strict policies, such as: using machine generated passwords, audit reports for users and administrators, and the ability for users to change their own password. [DOD85A]

d. *Monitoring On-Going Activity*

After the initial security checks have been conducted and an effective password selection policy is in place, the TCP Wrapper program should be configured to monitor on-going activity. The program will monitor vulnerable network functions like ftp, telnet, finger, tftp, etc. and record activity involving those aspects of the system. The program should be configured so that it detects host computers trying to use another host's name. Attempts of this nature should be logged and the connection immediately dropped. Logs of all activity should be stored in a protected archive. Consideration must be given about how much of the information is needed for archive purposes, since TCP often generates a large volume of information. The administrator can also set up methods to search the logs for suspicious activity. For example, a script file using the **grep** program can be used to search for key words or phrases. These activities can then be analyzed in more depth in order to ascertain if the system has actually been compromised.

Logs can provide critical information if a compromise of the system is encountered. Reconstruction of activity surrounding a successful attack can help the administrator find out how the compromise was accomplished, so that patches can be developed to thwart future attacks. Furthermore, these logs are critical pieces of evidence which may be used to prosecute attackers.

e. *Active Security Checking*

Ice Pick is an active program for testing network security. It is a different type of tool and should be managed differently. Ice Pick actively tries to compromise the security of a target system and reports the results to the security manager or administrator, as required. Such a program is essential in finding security weaknesses in the computer networks of an organization, but the program must be strictly controlled and not released to the general public. A program of this nature could be used by an attacker to launch automated attacks on any system connected to the Internet.

The best role for a program like Ice Pick is in providing a means to test the security of multiple systems on a network. An organization's security manager would need to coordinate active security testing with the system administrators. If the program is able to compromise a system, procedures must be in place to immediately correct the vulnerability. A primary goal is to use Ice Pick as a tool to assist network administrators in keeping each network secure and improving security practices on the network. Ice Pick can be used to target systems on a regular basis and report the results, providing a feedback mechanism for the administrator. This method of security testing, if used properly, can improve the security posture of the organization and instill an increased awareness of the importance of security.

C. MAINTAINING THE TOOLBOX

It is this author's opinion that DOD should be in the forefront of developing computer security tools and techniques. One particular area in which DOD should take a leading role is in the development and maintenance of security toolboxes for networked systems. This paper addresses the need for a toolbox to sunport Unix networks only, but the ultimate goal within DOD should be the development of a separate toolbox for each major system. If development of a toolbox for each system is not possible, then consideration must be given to minimizing the types of network systems allowed.

Even though the National Security Agency is responsible for development of secure network architectures, the reality is that it will be several years until all unclassified networks are replaced with newer and more secure implementations. A failure to encourage a toolbox standard will degrade network security and hinder efforts to protect sensitive information residing in distributed databases. Responsibility for maintaining security toolboxes should be given to an organization such as the Defense Information Systems Agency (DISA); specifically, the Defense Information Systems Security Program (DISSP) branch of DISA. DISSP seems like a logical choice for the consolidation of computer security tools and expertise within DOD.

Tools should be acquired, evaluated, and managed by dedicated personnel who understand security issues and how these tools can assist the administrator or security officer. The acquisition and maintenance of security tools will require close liaison with all government and commercial computer security organizations in order to provide the highest quality services and products. Once tools are obtained and evaluated, they should be placed on a secure system, which can be accessed via the Internet or a bulletin board, so that administrators can obtain the tools with the assurance that they have not been altered by unauthorized personnel. Use of a public key cryptographic system would be one method of distributing the tools to authorized administrators. A key element in maintaining a toolbox is that DOD administrators should be able to use DISSP as a central point of contact for computer security related tools, software, advice, and assistance.

To indicate the critical importance of security tools, one need only look again to the Internet Worm incident. The worm was detected soon after it was released, but it took quite a while before the code was analyzed and the method of attack was known. The worm used a password attack mechanism within the code as the means to gain access to vulnerable systems. Had all system administrators used a security toolbox containing programs to enforce strict password policy, ensure good password selection, and close known security holes, the damage caused by the Internet worm would have significantly decreased. Investment in the time and resources to maintain a security toolbox is cost effective. When deciding to implement a security toolbox such as that advocated in this paper, it should be remembered that the cost of recovering from just one incident like the Internet Worm was estimated from $200 to $53,000 at those installations that were affected. Higher estimates for recovery have been approximated for incidents involving malicious viruses. [REYN89] [KEHO92]

A closely related issue to the establishment and maintenance of a security toolbox is the control of those tools. Tools must be freely available for personnel involved in system administration, security, or integrity, but those personnel should realize that unscrupulous persons can also gain access to many of the same tools. Only by proper use of security tools and strict configuration control can attacks be mitigated or rendered ineffective. Passive tools such as COPS, SPI, Tripwire, TCP Wrapper, Passwd+, etc. provide information on system vulnerabilities that can ensure significant protection, if those vulnerabilities are corrected. Control and protection of passive tools is necessary, but controlling access to active tools is of an even greater concern.

Control of active security tools, such as Ice Pick, must be carefully considered, since these tools can be used in direct attempts to compromise some portion of a system. The results of the program feedback can immediately be used for illicit purposes. Such tools should not be freely distributed, due to the potential harm that could be caused by an attacker using the tool. It would be reasonable for a security organization to retain control of active tools and conduct system penetration operations remotely from a secure location. It is important that any holes found in the target system be immediately closed, since the information would probably be reported over a non-secure communications channel. An alternative method is to use a laptop computer and connect directly to the system to be tested. Security specialists could then travel to the site and test the system without the concern of broadcasting security holes found by an active tool.

Maintaining security toolboxes and providing advice and assistance is a critical need for protection of sensitive information collected, stored, and used by the government. Managers of computer networks need a single point of contact from which to obtain consistent guidance and new security tools. A single organization should be responsible for providing computer security expertise. This does not preclude development of a "virtual computer security organization" once distributed computing is more widespread, but security implications brought on by rapid technological changes cannot be dealt with independently within each government organization. This method is in use today, and it does not appear to be a good use of limited resources.

VI. CONCLUSION

A. SUMMARY

In the author's opinion, the critical nature of computer security, with respect to actual operations and procedures in the field, has not been dealt with adequately. There are good reasons for this lack of impact at the operational level. Recent advances in technology and proliferation of interconnected computer networks continue to challenge our ability to keep sensitive information secure. DOD, and NSA particularly, has been primarily interested in the protection of classified information for many years, but the capability to connect virtually every organization via computer network is introducing significant security concerns. Once organizations are interconnected, the key issue becomes one of the protection of sensitive unclassified information residing in the various databases. The future will bring a continued transition from older forms of communications to massive computer networks carrying electronic mail, text files, video, audio, and any combination of those forms of communication, and without some sort of central clearing house for computer security, the ability to respond to changes in technology will be degraded.

Computer security in the future will be problematic at best, but security will be virtually impossible if the investment in new security techniques is not increased. Maintaining an up-to-date security toolbox will be critical in responding to attacks or in preventing compromise of a computer network. It is the author's opinion that without the organizational capability for evaluating security tools and providing access to those tools, administrators will not be able to keep up with new security threats and methods to protect their system. The proliferation of computer networks will only increase in the future, and automated tools will be the only means available to protect information in organizations which have limited resources. The author believes that through centralizing much of DOD's computer security expertise and by using automated security tools, a greater return on investment can be achieved; however, this recommendation should be studied in more detail before action is taken.

The first goal of this thesis has been to emphasize security problems associated with government-owned Unix networks and present a recommended solution for enhancing the security of those unclassified Unix networks. The secondary goal has been to give an overview of the various organizations which are responsible for different aspects of computer security, and recommend changes which can facilitate expanded access to computer security technology and expertise. The two aspects of this thesis are closely related, and success in one depends on the other. Two of the key problems highlighted by the thesis and the recommended solutions are capsulized below:

***PROBLEM*:**

The utilization of Unix networks to manipulate and store unclassified sensitive information poses a significant problem for administrators and security officers. The problem is the most critical on those systems connected to the Internet.

***RECOMMENDED SOLUTION*:**

The creation of a security toolbox is recommended. The toolbox should contain automated security tools which will assist the Unix administrator and security officer in ensuring network integrity and security. The automated tools proposed for the initial version of the toolbox include COPS. SPI/Unix, Ice Pick, Tripwire, TCP Wrapper, and Crack.

***PROBLEM*:**

There is no management function or coordination method within the government that maintains and validates automated security tools. Also missing is a function or coordination method that provides consistent advice and guidance on computer security problems. The government organizations involved in computer security provide a vital function, but the scarce resources are scattered throughout various departments and agencies.

***RECOMMENDED SOLUTION*:**

Designate DISA, preferably the DISSP division, to perform the coordination function with respect to computer security issues at the operational level. The organization could assume the responsibility of validating and maintaining automated security tools for government administrators and security officers, as well as providing operational advice and guidance.

B. FUTURE RESEARCH: SHOULD GOVERNMENT COMPUTER SECURITY EXPERTISE BE CENTRALIZED?

1. Current Status

In this thesis, the author has focused on the need for a Unix security toolbox and for an organization which can advise, assist, guide, and manage security tools. As the defense budget is reduced, it will be necessary to decide whether it is necessary to centralize a majority of computer security expertise (for unclassified systems) in one organization. NSDD 145 gave NSA the responsibility to manage INFOSEC, but the question of how best to utilize scarce personnel resources has not been addressed. It may be time to decide if there is a better organizational structure and process for managing computer security throughout the government. Though necessary to maintain computer security expertise, making changes to organizational structure and management processes will be a tremendous challenge for the future.

ARPA has been a leader in responding to and funding the development of new technology which made networks such as the Internet a reality. The technology supporting vast networks like the Internet has now matured to the point that DOD must consider the security vulnerabilities of unclassified government-owned networks. With network technology constantly evolving and the resources shrinking, centralization of computer security expertise must be considered as an alternative.

In the author's opinion, though INFOSEC in DOD is centrally managed by NSA, the actual results of this management is not felt at lower levels. Each of the military service components must still develop computer security expertise individually, without the benefit of a well established computer security training program or professionalization standards. Indeed, the guidance for the Executive Branch of government has been that all departments and agencies develop a comprehensive Information Security (INFOSEC) policy for office automation systems. Each organization must set up a training program for security officers, administrators, and users. The general guidance does not stipulate at what level within each organization that computer security responsibility should be managed. [NTIS87] Part of the problem is the inherent difficulty of developing and executing policy in an area of expertise that must constantly evolve to keep up with technology. Coping with constant change is a difficult barrier to overcome in hierarchal government organizations.

The government attempts to retain as much in-house expertise as possible. Unfortunately, the speed of advancing technology has outpaced the ability to keep trained and experienced personnel, familiar with computer security issues. The troubling aspect of current practice is the standard to which policy is developed and personnel are trained. Who trains the trainers? It is well know what kind of training it takes to make a qualified military pilot and the other specialized training needed for specific airframes and operational scenarios. The proof of the training is demonstrated during flight qualifications and eventually in combat. What kind of training and certification qualifies a computer security professional and who should be qualified to set up an information security program?

2. The Need for Professional Standards

There is a need for professional standards so that competency can be measured, and this issue has not yet been resolved within DOD. Indeed, the topic of professional certification for computer science specialists has been quite contentious because of the difficulty in prescribing the standards. [CONG89] In the meantime, no standards are in place and technology continues to advance.

It is not adequate to simply direct that security policy be implemented. Without professional standards, the quality of INFOSEC programs will vary by organization. This type of fragmented approach hinders the ability of DOD to respond to the challenges posed by increased technological capability. Professional standards would ensure a specific level of expertise, but the only cost effective means to keep enough qualified computer security personnel is to centralize computer security assets in one organization. The critical issue is in the creation of a professionalization program and then determining how to manage career paths to ensure the military personnel remain competitive. Such a program could be based upon other professionalization programs such as the program used by the National Security Agency.

A proven method to guide the training and retention of top quality personnel is to set professional standards and give incentives for their attainment. A professionalization program would ensure that computer security personnel attain a minimum level of expertise which can be documented. A core set of requirements would need to address the basic knowledge and experience needed in the relevant areas of expertise. That basic knowledge would be a general professionalization standard, along with a required experience level (i.e. 2-5 years). Particular aspects of professionalization, could then be treated like a specializations, with a requisite amount of additional training and experience. It wouldn't be practical to initially train a person to be an expert in all relevant aspects of computer security due to the time required. Additional training, education, and job experience would qualify a person in specific areas of expertise.

3. Managing Expertise

In the author's opinion, the best short term solution to encourage professional standards and enhance security expertise would be to realign an existing organization, using billets from the military services and other DOD organizations that currently have the personnel. The best choice for realignment would appear to be the Defense Information Systems Agency (DISA) which is currently undergoing organizational change. The best location for computer security professionals within DISA would be the Defense Information Systems Security Program (DISSP). DISSP is a relatively new organization within DISA and it is currently attempting to define an expanded role in DOD INFOSEC.

The charter of DISSP should be reevaluated with a consideration of the following structure. Due to the critical nature of computer security and INFOSEC, especially for military purposes, the organization should be a semi-autonomous arm of DISA, with a military officer (at least O-7 equivalent) as commander and with a civilian deputy. The military commander position should be rotated between the services to ensure computer security issues are evenly addressed for all services. The civilian deputy should be appointed by the head of DISA and should serve a longer term than the military commanders in order to maintain continuity of policy. Computer security personnel from the military services should be incorporated into DISSP, and the organization should have the authority to conduct liaison with any government organization of any department when dealing with operational computer security issues.

DISSP should promulgate computer security standards for the government (focusing on sensitive unclassified information), with the assistance of NIST and NSA, and to provide advice and assistance for all government agencies on computer security matters. (NSA would retain its responsibility for classified information.) An organization like DISSP would gain expertise from having a professional cadre of computer security personnel, and it would provide a consistency in policy by the reduction in duplicative effort among other organizations.

Civilian personnel in the organization would provide long-term continuity, while the military personnel would bring in operational experience with which to improve security techniques. The military will gain from experience obtained by its personnel during an assignment as a computer security specialist. As military personnel are trained and gain experience in such an organization, they will take with them a much better understanding of computer security issues into future assignments.

4. Tomorrow

A centralized computer security "brain trust" is needed. This can be accomplished in the near term by physically relocating personnel to a single organization, but it may also require a completely new way of thinking about organizations. A *virtual organization* may well be what is required, eliminating the need for physical co-location of computer security experts. Once distributed computing becomes more widespread, *virtual organizations* can be created, which could yield the result of having all government computer security experts available to solve problems and provide guidance. This type of organizational concept will require new management techniques, coupled with an in-depth understanding of information technology.

5. Research on Managing Expertise and Standards

The author proposes the development of professional standards and new methods of managing computer security expertise. These issues are critical for the protection of sensitive unclassified information and a plan of action should be implemented. Further study is recommended to explore new management techniques which can use technology to synthesize and coordinate computer security within DOD.

APPENDIX

This appendix should be used to provide amplifying information for concepts discussed in the body of the thesis. There is some terminology used in the preceding chapters which will not be readily understood without an acquaintance with Unix security. The Unix operating system is complex and unless the reader has worked extensively with the security mechanisms in Unix, some concepts will require further reading. The appendix attempts to focus on those areas of Unix security which need to be emphasized for this paper. It is important to point out that there are many sources of information on Unix Security, and good security practices will require a familiarity with a wide variety of sources.

The information in the following chapters was obtained from several sources. The primary source was "Practical Unix Security" by Garfinkel and Spafford. [GARF91] The order of information presented was adapted from that of the book to highlight particular concerns relevant to this paper. Information from other sources is added where deemed appropriate, and indicated by references.

I. UNIX

The Unix operating system "...was not designed from the start to be secure. It was designed with the necessary characteristics to make security serviceable." --a quote from Dennis Ritchie, one of the original developers of Unix.

Unix does provide memory protection and access controls, but because Unix began in the "open" environment of universities and research institutions, security was not a critical factor in the development process. The security that was included in Unix was more to protect the operating system from being "crashed." The design of the operating system allows flexibility in running programs, but also leaves open the possibility that a program flaw will cause "security holes." The potential for "introduced flaws" dictates that the system must be monitored closely in order to ensure "security holes" are not allowed to compromise the system.

Unix was not developed to operate while connected to outside computer resources such as Wide Area Networks or Distributed Networks. Subsequently, the proliferation of communication lines and networks have increased the security threat to Unix systems. Since the advanced networking capability for Unix was developed later than the original program, security was not an integral part of that development.

II. PASSWORD MANAGEMENT

A. USERS

Before a user is assigned a password, the first consideration should be the choice of an appropriate username. There are several important factors which should be considered when assigning usernames. The username is only part of a Unix account which consists of the username and password. Because the username is an identifier, all usernames should be at least three characters long. Usernames should all be unique names and Unix special account names should not be used. Users can have more than one account and more than one username.

B. PASSWORDS

The password is an authenticator which should not be displayed on the system. An account can be locked out after a preset number of unsuccessful password attempts. This capability is a good security measure, because the system administrator must then unlock the account. If allowed, the password can be changed by the user with the **passwd** command.

1. /etc/passwd

The **/etc/passwd** file maintains unique identifiers for all users on a Unix system. The file "...contains the username, real name, identification information, and basic account information...". The following example shows the fields used to identify each account and Table 1 gives the meaning of each field.

i.e. **brown:eZ3/.bs7BZ3dx:111:110:Tom Brown:/u/brown:/bin/csh**

TABLE 1: EXAMPLE OF "**/etc/passwd**" FIELDS

FIELD	CONTENTS
brown	Username
eZ3/.bs7BZ3dx	Encrypted Password
111	User Identification Number (UID)
110	Group Identification Number (GID)
Tom Brown	User's Full Name
/u/brown	User's Home Directory
/bin/csh	User's Shell

[GARF91: p. 23]

2. Encrypted Password System

Only encrypted versions of passwords are stored on the Unix system. **Crypt(3)**, a one-way function, is used to encrypt a password before it is stored on the system. It is based on the Data Encryption Standard (DES), and the result of **crypt(3)** is eleven printable characters. The eleven characters provided by **crypt(3)** are the characters actually stored in the **/etc/passwd** file.

Upon login a user password is taken by **/bin/login** and encrypted with the one-way function. The result of the function is then compared with the **/etc/passwd** version of the password.

a. DES SALT

The SALT is a 12-bit number used by DES to change the result of an encryption. The **/bin/passwd** program selects the SALT. [GARF91: p.31] The SALT is then converted to a digraph and stored with the encrypted password.

TABLE 2: ENCRYPTED PASSWORD FORMAT

Password	SALT	Encrypted Password
ball5551	aM	aMdjtT44FRqop
ball5551	8R	8RcuizX32G4dq
duty$FREE	33	33Mp*/u76GSQg

Note: If two users choose the same password, each password will have different representations in the password file.

3. The System Administrator's Role

Password security is the first line of defense for many Unix networks. There are several actions which can be taken by the administrator to ensure a strong password system. One consideration for the administrator is whether to use a password selection or checking program to assist users in choosing secure passwords. Password generators have also appeared; however, users do not normally like to use such programs because they wish to choose their own password.

4. Shadow Password File

In order to make it more difficult for attackers to exploit password vulnerabilities, shadow password files are used so that the encrypted passwords cannot be read by users on a system. This type of file is used to store encrypted passwords separately (sometimes with other security data) from the additional information resident in a normal password file. The shadow password file is set so that only root can read the file, offering more protection from attackers. Many programs require user information but not the password; therefore, they can use the normal password file to retrieve needed data. [FERB93] Unix systems classified as C2 have a shadow password capability. Administrators should also ensure that there are no extra copies of /etc/

passwd residing on the system in a location that is readable by the public or by unauthorized users.

5. Password Aging & Login Restrictions

Some systems allow the administrator to limit the life of a password, forcing the user to change passwords on the expiration date. Password aging and expiration can enhance system security by ensuring old passwords do not remain on the system. Users should be made aware of the purpose of changing passwords, and advised to change their passwords on a periodic basis.

Users are sometimes required to travel, leaving the account basically dormant. When a user will not be using the system for a long period of time, it may be appropriate to disable login to that account until the user returns. Disabling the login is a simple procedure which is done by placing an asterisk [*] before the first character of the password in the **/etc/passwd** file. [GARF91: p. 41]

ex.**brown:*23dFD5cv8mK:181:100:John Brown:/n/brown:/bin/csh**

III. USER IDENTIFICATION

A. USER IDENTIFIER (UID)

The UID is a 16 bit number ranging from -32768 to 32767 or ranging from 0 to 65535. The numbers 0-9 are reserved for the system, with user numbers beginning at 20 or 100. The UID is kept in the **/etc/passwd** file, after being assigned to a user.

The system actually uses the UID to identify the user, not the user name. Two users with the same UID are considered to be the same user by Unix. The UID is the unique identifier and not the login name or password. It is not a good practice to assign two users with the same UID.

B. GROUP IDENTIFIER (GID)

Each user on the system must belong to at least one group. Groups are assigned a name as well as a GID. Upon creation of a user account, the administrator assigns the GID and name. Each group is allowed specific privileges on the system, such as read, write, and/or execute capability for specific files, directory structures, or peripheral devices.

The GID of the primary group associated with a user is stored in the **/etc/passwd** file. Groups provide the administrator another method of controlling access to the system, specific applications or other areas which require protection.

1. /etc/group

A database of group information, similar in format to the **/etc/passwd** file, is stored in the **/etc/group** file. The most critical group is the "wheel" group, which normally refers to a listing of all the system administrators. The "users" group may be assigned for all users on the system.

The primary group for each user will be the group number assigned in the / **etc/passwd** file. Users can be assigned to other groups in addition to the primary group and those groups are handled differently according to the version of Unix being used.

Example of **/etc/group** entry: **wheel:*:0:root, brown, alex**

TABLE 3: MEANING OF "**/etc/group**" FIELDS

FIELD	CONTENTS
wheel	Group name
*	Group's Password
0	Group Identification Number (GID)
root, brown, alex	Users assigned to the group

[GARF91: p. 47]

2. System V Unix

In System V Unix, the user can belong to only one group at a time. The "**newgrp**" command is used to change the users current group and the user is allowed to change groups, without regard to whether the user is a member of the group. If the user wishes to change groups but is not a member of the group requested, the system will request the group password. If the correct password is provided, the user is placed in the group temporarily, acquiring all privileges of that group.

The password for a group is calculated in the same manner as the normal user's account password. Most systems do not have programs to change group passwords.

3. Berkeley Unix

In this version of Unix, the users may belong to more than one group, simultaneously. The **/bin/login** program is used to automatically scan the **/etc/group** file. Once scanned, the program places the user in the particular groups where he or she is authorized. The **"newgrp"** command, used in System V, is not used in Berkeley Unix.

C. SPECIAL USERS/PROGRAMS

1. root

Root i‿ the most important special user, also called the superuser. The superuser has their own password called the "root password" and the UID of root is "0". The root account provides "housekeeping" functions which provide complete control over the operating system.

There are no security checks of root operations when the superuser runs a program, which highlights the importance of protecting the root password. Because of the powerful nature of using root privilege, the superuser should not use the root account as a personal account. The root account should only be used to perform system checks or system administration duties.

The **su** command is used when the administrator needs to become superuser and perform administration duties. The **su** command logs the time and user identification when it is run, providing an audit trail of users in the root account. The normal procedure for the system administrators should be to log in under their own user account and then use the **su** command to perform duties in the root account.

Despite significant capabilities, the superuser does have a few restrictions placed upon their activities. The superuser cannot make changes to a file system if it is mounted as Read-Only. Such a filesystem would have to be unmounted and then remounted as a Read-Write mode filesystem. Another limitation is that the superuser cannot decrypt stored passwords (assuming non-use of the **crack** program). The superuser can, however, record passwords as they are typed in by the user. The superuser cannot terminate a process that has entered a wait state in the operating kernel. The only recourse for the superuser would be to shut down the system, which kills all processes.

Because the superuser is the most powerful user with all privileges, it is this power that is the main vulnerability in Unix system security. Unix attacks often comprise the efforts of an attacker to become the superuser. Also, many of the security holes in Unix have been associated with the process of regular users obtaining superuser privileges.

2. Unix-to-Unix Copy Program (UUCP)

UUCP is the portion of the system which can transfer files and electronic mail between systems which are connected via phone lines. The remote computer must login as **uucp** when it calls another Unix system. Email that is to be transmitted is stored in directories that cannot be read by users, but only by **uucp**. UUCP vulnerability will be discussed in more detail in subsequent sections.

D. SUBSTITUTE USER

The command **su** is an abbreviated form of "substitute user". It is used to change a user's ID on a temporary basis. The format below shows how a system administrator gains root access from his or her personal account:

```
% /bin/su
% password: IamRoOt
#
```

<After providing the correct password, the user now has root privileges, as denoted by the pound sign at the prompt.>

The **/bin/su** vice **su** command should be used by the administrator when becoming the superuser, because using the entire path helps protect against Trojan

Horses. Recent versions of Berkeley Unix require a user to be in the wheel group before they can use the **su** command to become root. Some versions of **su** allow users in the wheel group to use their personal password to gain superuser access. Attempts to gain superuser access using the **su** command are logged in the **/usr/adm/messages** or **/usr/adm/sulog** (System V) file.

IV. FILES

Unix has a structured file system, which from a very basic point of view includes files and directories. Directories contain files, but they also contain devices (with logical names), symbolic links, and other directories in a hierarchal arrangement.

The **ls -lF** command gives a more detailed listing of files in a directory as indicated below. In System V, the group name replaces the user name when using **ls -lFg**. Table 7 gives the meaning of the display generated by the **ls** command shown below.

i.e. **-rwxrwxrwx 1 brown 606 Sep 23 23:59 unixinfo/**

Table 8 shows the different symbols that can appear at the end of a file name.

There are three times associated with a file: **mtime**, **atime**, and **ctime**. The **mtime** and **atime** can be easily changed by a system library routine but the **ctime** normally cannot be changed and reflects the last time the file is written, undergone a protection or owner change. It is important to note that the **ctime** can be changed if an attacker obtains superuser access and changes the system clock.

ls -l includes the modification time of the file. (**mtime**)

ls -lu includes the last access time of the file. (**atime**)

ls -lc includes the inode change time. (**ctime**)

TABLE 4: EXAMPLE OF "**ls -lF**" OUTPUT FIELDS

FIELD	CONTENTS
-	File Type. (A dash denotes a file.)
rw-r--r--	File Permissions
1	Number of Hard Links of the file.

TABLE 4: EXAMPLE OF "ls -lF" OUTPUT FIELDS

FIELD	CONTENTS
brown	File owner.
606	File size, in bytes.
Sep 23 23:59	Time the file was last modified.
unixinfo	File name.

TABLE 5: FILENAME SYMBOLS

SYMBOLS	MEANING
(blank)	Regular File.
*	Executable Program or Command File.
/	Directory
=	Socket (BSD only)
@	Symbolic Link

[GARF91: p. 59]

A. FILE PERMISSIONS

1. Read, Write & Execute

The file type is indicated by the first character of the file permission. The various file types are shown in Table 6. [GARF91: p. 61] The characters which follow the file type indicate read, write, or execute permission for owners (1st group of 3 characters after the file type), groups (2nd group of 3 characters), and others (3rd group of 3 characters) respectively.

Example: **drwxrwxrwx** <This entry before the directory name indicates that the owner of the directory, groups, and others all have the ability to read, write, and execute any files which are in that directory.>

TABLE 6: FILETYPE SYMBOLS

SYMBOLS	MEANING
-	Plain File
d	Directory
c	Character Device (ex. printer)
b	Block Device (ex. disk or tape)
l	Symbolic Link
s	Socket (Device)
p	FIFO (System V only)

2. "chmod"

a. Changing File Permissions

Example format: **chmod [-R] [agou] [+-=] [rwxXstugol]** ***filename***

Example entries: **% chmod o+w** ***myfile***

% chmod o+r ***myfile***

% chmod -R o+x ***mydirectory***

TABLE 7: FILE PERMISSION SYMBOLS FOR "**chmod**"

SYMBOLS	MEANING
-R	Recursive **chmod** (BSD only)
a,g,o,u	Modifies All, Group, Other, or User permissions
+, -, =	Adds, Removes, or Replaces current permissions
r,w,x,X,s,t, u,g,o,l	(r)Read, (w)Write, (x)Execute, (X)Execute only if the file is a directory, (s)SUID/SGID, (t)Sticky Bit, (u)Takes permissions from user's permissions, (g)Takes permissions from group's permissions, (o)Takes permissions from other's permissions, (l)Enable's mandatory locking on file (System V only)
filename	filename can be a single file or a group of files

b. Octal File Permissions

The **chmod** command may also be used with 4-digit octal representations for file permissions.

TABLE 8: GENERIC OCTAL FILE PERMISSIONS FOR "**chmod**"

OCTAL NUMBER	FILE PERMISSION
0400	Read by Owner
0200	Write by Owner
0100	Execute by Owner
0040	Read by Group
0020	Write by Group
0010	Execute by Group

TABLE 8: GENERIC OCTAL FILE PERMISSIONS FOR "**chmod**"

OCTAL NUMBER	FILE PERMISSION
0004	Read by Others
0002	Write by Others
0001	Execute by Others

[GARF91: p. 67]

Example entry: **chmod 666 *.txt** (Allows the file owner, groups, or others to read or modify any file and with the extension "**.txt**")

B. "UMASK"

The **umask** (user file-creation mode mask) uses a three-digit octal number to set the default file permissions for a new file. The umask is set in the .login, .cshrc, or .profile files. **Umask** is a built-in function which is available in various shells such as **sh, ksh, csh.** The bits that are set in the umask correspond to the permissions that are not set automatically upon creation of a new file. This is accomplished by using a bitwise complement procedure.

The default for most Unix utilities is 666 (any user may read or modify the file). The default for most Unix programs is 777 (any user may read, write, or execute the program)

Examples of how **umask** is used:

0666 (default file creation mode denoting that any user can read/write to the file)

0022 (umask octal number)

0644 (resulting permission, based on combining 0666 and 0022)

TABLE 9: "**umask**" RESULTS (BASED ON FILE DEFAULT OF 0666)

umask	User Access	Group Access	Other
0000	all	all	all
0002	all	all	Read
0007	all	all	None
0022	all	Read	Read
0037	all	Read	None
0077	all	None	None

[GARF91: p. 71]

C. SUID (Set-UID) and SGID (Set-GID)

The SUID/SGID program is a program that can assume another UID/GID when it runs. When such a program is run by a user, the program obtains the permissions of the user who created the SUID/SGID program. Most SUID/SGID programs are **root** programs, which means that the program assumes **root** privileges when it is run. SUID/SGID programs can be the cause oı serious security holes if they are not properly maintained. The reason for vulnerability stems from the fact that if an attacker can manipulate the operating system to run a command as **root**, the attacker can trick the system into allowing the attacker **root** access.

If the system uses mounted file systems, especially remote file systems, the SUID/SGID programs should be disabled. Disabling can be accomplished using the **nosuid** option with the **mount** command. The primary reason for using **nosuid** is to ensure that SUID/SGID files are not imported from the mounted filesystem.

Example: **/etc/mount -o nosuid gemini:/grus /usr/grus**

D. DEVICES

Devices are treated like files in the Unix system. The O/S kernel changes program reads and writes to an I/O operation when a device file is specified.

Sample of special devices:

/dev/null (trashs anything written to this device)

/dev/console (writes to the console anything written to the file)

/dev/kmem (reading & writing to this device accesses the kernel's memory)

Some devices should be world-writable, such as **/dev/null**, **/dev/tty**, and **/dev/console.** Most other devices should not be readable or writable. The **/dev/kmem** device is particularly vulnerable to an attacker if it is not properly protected. All access to the O/S kernel's memory should be strictly controlled. A security checklist should include checking the status of all devices.

E. FILE OWNERSHIP

The **chown** command is used to change file ownership. In Berkeley Unix, only the superuser can change file ownership, as compared with System V in which the user can execute the **chown** command to change ownership. Changing ownership is a powerful command, and can lead to security problems in System V. Problems are possible when an attacker gains access to a system because the **chown** command can be used to hide the evidence of their presence.

F. GROUPS

The **chggrp** command is used to change the group access to a file. Only the file's owner, who is a member of the group to which the file is to be changed, or the superuser can change a file's group in Berkeley Unix. In System V Unix, the owner may change the group of any file owned to any other group. The potential vulnerability is obvious.

G. IMPORTANT UNIX FILES

TABLE 10: COMMON SUID PATH AND FILE LISTINGS IN BSD UNIX

PATH	SUID FILES
/usr/etc/	ping; timedc; rdump
/usr/lib/	sendmail; lpd; ex3.7recover; ex3.7preserve
/usr/lib/x11/	getcons
/usr/lib/uucp/	uucico; uuclean; uuxqt
/usr/bin/	mailq; newaliases; iostat; atq; at; atrm; tip; cu; login; su; uucp; uux; uulog; uuname; uusnap; uupoll; uuq; uusend; ruusend
/usr/bin/x11/	xterm
/usr/ucb/	lpr; lpq; lprm; w; uptime; quota; rcp; rdist; rlogin; rsh; talk; chsh; chfn
/usr/var/uucp/	uumonitor; uucompact; uumkspool; uurespool

V. UNIX ACCOUNTS

A. ACCOUNT MANAGEMENT

The password is a critical point of vulnerability for an account and those accounts without passwords should be considered a crucial security violation. It is possible to use the **awk** command to search for missing account passwords in order to detect weak points in the system. i.e. % **awk -F: 'length($2)<1 {print $1}' < /etc/passwd** [GARF91: p. 90] Different systems may have passwords located in files other than **/etc/passwd,** therefore, the administrator should check the documentation for the particular version of Unix being used, to determine search parameters. The table below shows the meaning of **awk** commands used for conducting searches of files. [SOBE89: p. 411]

TABLE 11: EXAMPLE OF "**awk**" COMMAND

FORMAT	MEANING
awk	Utility in Unix which performs pattern scanning and report generating.
-F:	Indicates that the Field Separator is a colon ":" .
length($2)<1	Specifies that the utility is to scan the 2nd field and check to see if it is less than 1.
{print $1}	Specifies that the first field of each entry searched is to be printed.
< /etc/passwd	Specifies the file which is to be scanned.

Many vendors have shipped systems with default account passwords or with no passwords at all. The administrator should always check a new system for accounts which may already be assigned a default password and change the password immediately. Another option is to disable the account by placing an asterisk in the password field. A good starting point is to first check the **/etc/passwd** file to see what accounts have been set up by the vendor. It is also important to note that some software applications establish an account and password upon installation. These passwords should always be changed after the installation process is completed.

Some special accounts only run a single command and may not have a password. ex. **finger, date, who**, and **sync**. Single command accounts should not be allowed to have any *shell escapes*, nor should they take any sort of keyboard input. The goal is to ensure that such accounts cannot be made interactive with the user, because an attacker could exploit this weakness.

Open accounts have names like **open** or **guest** and may not require passwords. Such accounts are weak points in system security, where an attacker may try to gain access to parts of the system other than the authorized work area. Open accounts are not recommended, but if they are necessary, password protection should be seriously considered.

This type of shell can be used to limit the vulnerability of an open account. The restricted shell "**rsh**" (System V) first executes commands in the **$HOME/.profile** file which restricts the user to the current directory. The user cannot use command names that contain a slash, cannot redirect output, and cannot change the path. The administrator can also ensure that the user cannot use the account over a network. BSD Unix uses the **sh** program to create the **rsh** restricted shell, which has the same capability as under System V. Restricted accounts should not have applications that can run subprograms, since this capability provides another possible point of attack.

B. GROUP ACCOUNTS

Group accounts are those used by more than one user, using only one password. Group accounts are a potential security weakness because of the lack of accountability in determining who does what on the system, and the difficulty of controlling the password. An alternative to a group account is the creation of a group in the **/etc/group** file and adding each user to this Unix group. The user must first access the system with their own password and then access the group.

C. DORMANT ACCOUNTS

If a user is scheduled to be away for a significant period of time, the administrator should consider disabling the account. There are three basic methods to prevent unauthorized login to a dormant account, which include changing the password, disabling the password, or changing the login shell. A procedure to find dormant accounts is to use a shell script and the **last** command to find accounts that have not been opened for a significant period of time. Specific examples of shell scripts can be found in several texts. [GARF91: pp. 98-99]

D. ACCOUNT PROTECTION

The wheel group is used for all system administrators. Belonging to the wheel group allows the administrators the capability of using the **su** command to obtain superuser access. [FERB93]

Secure terminals are used to protect **root** access and make it more difficult for an outside attack designed to guess passwords. Unless the terminal is annotated as secure in the **/etc/ttys** file, the superuser is not allowed to login directly, using the root password. The administrator would have to log in as a regular user and then use the **su** command to gain root privilege. This procedure makes an attacker's job more difficult.

VI. SECURING DATA

The most critical aspect of any system is the Backup. Restoration of a system is straightforward and loss of data is a minimal problem if backups are conducted on a timely basis. All files and directories should be backed up, with particular attention to user files, system databases like **/etc/passwd** and **/etc/ttys**, and system directories that are very important such as **/bin** and **/usr/bin**. Backups of critical system databases is an essential security precaution. Databases such as **/etc/passwd** are very sensitive and files such as **/etc/rc** and **/usr/lib/crontab** can be modified and become a threat if an intruder gains access to the system.

Backup of critical files give the administrator the ability to recover from attacks on the system. Table 12 provides a listing of some of the important files.

TABLE 12: IMPORTANT UNIX FILES

FILE	PURPOSE OF BACKUP
/etc/passwd	File may be used to recover new accounts.
/etc/group	File may be used to recover new groups.
/etc/rc*	File may be used to recover changes in the boot sequence.
/etc/ttys or /etc/ttytab or /etc/init-tab	File may be used to recover terminal configuration changes.
/usr/lib/aliases	File may be used to recover changes in mail delivery.
/etc/exports	File may be used to recover changes in NFS filesystem security.
/etc/netgroups	File may be used to recover changes in network groups.
/etc/fstab	File may be used to recover changes in mounting options.

[GARF91: p. 110]

A. INTEGRITY

The goal of checking data integrity is to prevent unauthorized modifications and to provide a means by which an administrator can verify the integrity of system files. Modifications to files can result from many actions, from intruder attacks to inadvertent user activity. The system administrator should keep a checklist of files which should be regularly checked for integrity. Files such as **/bin** and **/usr/bin** are seldom modified and would be good candidates for regular integrity checks.

Using file permissions is the first line of defense for protecting data integrity. Software can also be used to enhance integrity by using the **mount** command to ensure appropriate disk drives are mounted in a Read-Only mode. This procedure normally cannot be circumvented unless an attacker gains **root** access. A disk may be write-protected to prevent inadvertent or unwanted writes, but this method complicates disk usage because all partitions are restricted.

Unix commands as well as programs like Tripwire can be used to check file integrity. The **diff** and **cmp** commands are used to compare files, in addition to the **rdist** command (BSD Systems) which can be used to compare files between two systems. **Rdist** may also be used to update target files or can be run by the administrator to update or check software on a regular basis.

B. LISTS & SIGNATURES

Using the **ls** and **find** commands, an administrator can write a script file to list the files and directories, as well as check parameters such as modification time, owner, group, size, etc. The results of the listing should be stored to a file, which can be compared with future utilization of the script.

A simple form of signature is to use the output of the list procedure above. Some versions of Unix have the **sum** command with will calculate a checksum for a file. A better solution would be to use a cryptographic checksum or a good hash function. Newer versions of Unix have a **des** command which can create a cryptographic checksum. Other more specialized programs such as Tripwire incorporate sophisticated cryptographic digital signature methods.

VII. LOG FILES

Log files keep a history of the commands executed by users and the times that the users login to the system. The audit system of Unix is based on the log files.

A. /usr/adm/lastlog

The last login time of each user is kept in the **lastlog** file and should be displayed each time a user logs into the system. The last login time may also be displayed by using the **finger** command. The lastlog information can assist in determining if an unauthorized user has accessed an account, especially if all users check there own accounts upon each login.

B. /etc/utmp & /usr/adm/wtmp

The **/etc/utmp** file records all users who are currently logged into the system and **/usr/adm/wtmp** records all logins and logouts. In BSD Unix, the information stored by these files include the username, hostname, terminal name, and the logon time of the user. In System V Unix, the information stored includes the username, device name, terminal line number, process identification, exit status of the process, a code for the entry, and the time of the entry.

The commands **who, users**, and **finger** use the **wtmp** file to provide the information requested. The **last** program is used to display all logins and logouts associated with every device on the system. The **last** program uses the **utmp** file for its information. It should be obvious that an attacker with access to these log files can eliminate information concerning his or her presence on the system.

C. /usr/adm/acct

This file can be used to log all commands from all users. This type of system accounting procedure is used primarily for security reasons and when CPU time is related to billing. The **/usr/adm/acct** must be frequently purged and put in a summary file due to the amount of data that can quickly accumulate.

The **accton** command is used to initiate system accounting and recording in the / **usr/adm/acct**. Upon termination of a process the Unix kernel records the username, command name, CPU time utilized, and the time the process was exited.

The **lastcomm** command is used to display the information in a readable format. The command should be executed only by the administrator, especially if it is used to monitor intruders on the system.

D. LOGS

In BSD Unix, the **openlog, syslog**, and **closelog** commands are used to record logs of system functions. The **/etc/syslog.conf** file is used to control the location of logged messages. The following example shows the format of the **/etc/syslog.conf** file. [GARF91: p. 133]

```
*.err; kern.debug; auth.notice
```

(Meaning: All error messages, kernel debug messages, and notice messages which are generated by the authorization system will be sent to the console. If the system console is defined as a printer, then the messages will be printed. A file name could also be added to the line above so that the output is saved to a file.)

VIII. THREATS TO THE SYSTEM

Programmed threats are those types of attacks that exploit a program's execution of instructions. Viruses are only one small portion of programmed threats, which include Worms, Trojan Horses, Bacteria, Logic Bombs, and Back Doors.

Checking new software and ensuring that new software is received from a trusted source is a very important protection measure. The damage caused from programmed threats can range from merely a nuisance to unrecoverable data losses. Protection from damage is the job of the system administrator and all of the users of the system. The system administrator must institute appropriate security safeguards for the system. Users must be made aware of their responsibility for following the security guidelines.

Unix attacks are primarily from trojan horses or back doors. The first places to look for such problems are in the shell, the start-up mechanisms, and automatic mechanisms. System attacks may intend to damage the system or to gain access to data or to the system itself.

Path attacks are those where the attacker inserts a shell script in a compromised directory, which could then execute a command in a different manner. If a system administrator changed to the problem account directory and executed a command, a shell script could be inadvertently initiated.

Start-up file attacks are possible because once applications are set up and the initialization procedures set, the start-up variables are rarely checked. Since initialization options are conducted automatically, the start-up file is an attractive target for an attacker. Start-up files should be protected so that the owner is the only person who can have write access to the file. Table 13 gives some examples of files which can be vulnerable to start-up attacks.

TABLE 13: FILES VULNERABLE TO STARTUP ATTACK

FILES	EXECUTION METHOD
.login, .profile, /etc/profile	Executed by the shell when the user logs into the system.
.cshrc, .kshrc	May be executed at login or at a later time.
.emacs	Executed when a user starts up the emacs editor.
.exrc	Executed when the **vi** or **ex** editor is started.
.xinitrc	Executed when XWindows is started.
.sunview	Executed when sunview is started.

An example of a possible attack method follows:

```
% cat > .exrc
! (cp /bin/sh /tmp/.secret;chmod 4755 /tmp/.secret)&
^D
```

(If an attacker placed the preceding commands in every directory where he or she has gained write access and if the superuser, at some point in the future, executes **vi** or **ex** in any of the targeted directories, the command above will create a SUID **sh** which is stored in a location specified by the attacker. The attacker can then use the SUID file and take advantage of the system using a superuser SUID file.)

A. PROTECTING FILES

Always check for "world-writable" files, and modify the permission when possible. If the **.rhosts** file is world-writable, an attacker could take over the account remotely over a network. World-writable files, directories, or devices should always be kept to an absolute minimum. Devices can be a particularly serious security problem. If a hard disk is made world-writable, an attacker can write files to the disk with a potential for compromising the entire system. Group-writable files can be also be dangerous and should be treated in the same category as world-writable files.

COPS and other security tools will assist the administrator in this function. The **find** command can also be used to find world-writable files, directories, or devices. An example of how to use the **find** command is shown below.

ex. **find / -perm -2 ! \(-type s -o -type p -o -type l \) -print**

<The example above prints all world-writable files, omitting sockets, pipes, and symbolic links. [GARF91: p. 167]>

B. SUID/SGID

A listing of all SGID/SUID files on the systen, should be maintained. This list will help in comparing authonzed from unauthorized files found by regular scanning of the system. An administrator should be wary of any SUID/SGID program that does not reside in the **/usr/etc, /usr/lib, /usr/ucb, /usr/bin, /bin,** or **/etc** directory. It is also important to note that a user who is allowed superuser access even for a short period of time could be able to gain superuser access in the future, as demonstrated by the example below.

```
% cp /bin/sh /usr/lib/.mysh
% chown root /usr/lib/.mysh
% chmod 4755 /usr/lib/.mysh
```

SUID shell scripts made by the superuser should never be used due to the fact that normal users can potentially gain superuser access. REMEMBER: "SUID shell scripts cannot be made secure"!

Reminder of how SUID programs work: The normal operation of a process is limited by the access privileges of the user of the program. A program can be made

SUID (Set User ID) by changing the permission with the command **chmod u+s.** Once SUID, any processes carried out by the program will have the same access permissions as the owner of that program. Therefore, if **root** creates a program and it is made SUID, any processes run by that program have **root** access privileges. If an attacker could alter such a program, any processes carried out would have total access to the system.[WOOD85: pp. 20-22]

IX. MODEM COMMUNICATIONS

A. SECURITY

Security is a concern when using a modem because the connection is normally outside of the location of the originator. Protecting the phone numbers associated with a modem is the first line of defense in protecting serial communications. A better protection mechanism is to have phone lines which only work in a one-way mode, either calling out or calling in. In conjunction with one-way lines, the modems can be segregated into call-out and call-in modems. By protecting the modems and phone lines, a potential attacker is prevented from using common attack methods such as "spoofing" the communications software of the target computer system. (Spoofing is a method which tricks the system into treating the distant computer as an authorized user.)

B. UNIX AND THE MODEM

Unix has the capability to send and receive data via modem communications. The **tip** and **cu** commands are used to facilitate communications with other computers, which could be using other operating systems. Using the UUCP capability, Unix can use modems to send or receive electronic mail. The SLIP (Serial Line Internet Protocol) or PPP (Point-to-Point Protocol) can be used with Unix and modems to connect remote computers with the network.

Installing modems on a Unix system will require modification of system files such as **/etc/ttys, /etc/remote, /usr/lib/uucp/L-devices, /etc/inittab** or **/usr/lib/uucp/ Devices**. Permissions for any device connected to modems should be set to 600 and owned by **root** or **uucp**. If the permissions are not set correctly, users may be able to read communications of other users on the system.

1. Testing the Modem

Upon acquiring modems for a Unix system, the first important action is to test the modem. Testing requires that the user communicate from another computer, calling the modem to be tested. [GARF91: pp. 186-192]

a. Procedures to Check Control Signals

(1) Use the **tip** or **cu** command to call the modem and try to login to the target computer.

(2) Disconnect the telephone line from the originating modem. The user should be returned to the Unix prompt.

(3) Call the modem once again and disconnect by turning off the sending modem. The user should be returned to the Unix prompt.

(4) Call the modem once again, exit from the **tip** or **cu** program and leave the modem connection in place. The user should be returned to the Unix prompt and the modem should hang up automatically.

(5) Call the modem once again, and this time use a third terminal to kill your **tip** or **cu** process. The modem should hang up automatically.

b. Procedures to Check Answering

(1) Call your computer. The login banner should print out on the screen. Login to the computer using normal procedures.

(2) Use the tty command to determine the line being used, then logout. The phone should be hung up by the computer.

(3) Call your computer again and login. Pull out the telephone line from the sending modem.

(4) Call your computer again. The computer should allow you to login and you should not have been placed in your previous shell. The old connection should have

been broken, otherwise there is a problem. (This procedure should be repeated after disconnecting the telephone line going into the sending modem.)

(5) Unix should logout a user anytime a connection is broken and if this does not happen, a security flaw exists in the system.

(6) All modems connected to the Unix system should be tested thoroughly.

(7) Modems connected to hunt groups should be checked to ensure proper operation.

2. Modem Security

(1) Protect access to the phone line itself.

(2) Disable call-forwarding on any line connected to a modem.

(3) Use a leased line when feasible.

(4) Ensure third-party billing is disabled. (This prevents persons from billing their calls to the modem line.)

(5) Use modems that require a password to verify that an authorized user is attempting access.

(6) Use modems that call back to the originator to verify the origin of the call.

(7) Use encryption.

(8) Use automatic number identification, if available.

X. UNIX-TO-UNIX COPY (UUCP)

A. WHAT IS UUCP?

UUCP provides the capability to send mail or news to users over a network, transfer files, and execute commands remotely. It can be used over serial communication line or telephone lines connected to remote networks. UUCP can also store and forward messages.

1. UUCP Commands

a. uucp

The **uucp** command copies files from one computer to another, with both computers using the Unix operating system. The program can also be used to transfer files from one remote computer to another, but the originating system must be connected to both remote systems. The normal configuration of **uucp** only allows copying into the **/usr/spool/uucppublic** directory. When copying files with **uucp**, the filename can be changed as it is sent.

b. uux

The **uux** command executes programs on remote Unix systems. It was also used to print files onto other computers, in the days before local area networks became a common method of manipulating and transferring data. The primary use today is sending mail between computers.

B. UUCP SECURITY

Security should be an immediate concern when a system allows programs or files to be copied from one computer to another. The **uucp** programs must be SUID, but they should be run as **uucp** SUID and not **root** SUID. Since **uucp** has no special privileges, the primary danger would be the possibility of an attacker being able to read files owned by **uucp** or create files in world-writable directories or directories owned by **uucp.**

The **uucp** account can be assigned a password to prevent unauthorized access. This protection should be considered by the administrator.

1. Security Provided by the System Administrator

A **/etc/passwd** entry should be established for each system that calls your computer. Permissions can be set according to the access required. This method allows much more control over how much access remote systems can have to your system. The directories to which remote systems may have access should be limited. Remote file retrieval can be denied altogether, if desired.

The administrator should require callback procedures to verify the originating system. Access should be denied to any computers which have demonstrated suspicious activity on your system.

2. UUCP Passwords

Changing the UUCP account password requires the superuser to use the **passwd** command and the account name. UUCP account passwords should be set immediately when a new Unix system is being set up. The **/usr/lib/uucp/L.sys** file contains telephone numbers, accounts, passwords, etc. of remote systems with which your system communicates. This file must be protected from unauthorized access. Information in the **L.sys** file should be unreadable to anyone except UUCP.

C. UUCP-VERSION 2

Five programs are used in Version 2 in order to improve security and control access to your system. The five programs are **USERFILE, L.cmds, SEQFILE,**

FWDFILE, and ORGFILE. USERFILE and **L.cmds** are two of the more important files.

1. /usr/lib/uucp/USERFILE

This program controls access to files which may be used by UUCP. It specifies which directories are authorized access by remote systems, the login name that the remote computer must use, whether callback is to be used, and which files may be sent over UUCP by users on the local system. USERFILE can be used to give additional UUCP privileges to specific users. The USERFILE should also have one file without a username and one file without a system name. These files are used by **uucico** or **uuxqt**. USERFILE entries have the following format:

```
smith, gemini c /usr/spool/uucppublic
```

TABLE 14: DEFINITION OF USERFILE FIELDS

ENTRY	MEANING
smith	User login that is used in **/etc/passwd**.
gemini	The remote system's name.
c	Callback Flag. (Optional)
/usr/spool/uucppublic	Absolute pathname. If left blank, open access is allowed to any file.

a. Local Users

A USERFILE entry can be made for all users allowed to transfer files, but this capability is limited to 20 entries. An alternative to specifying entries for each local user is to allow file transfer for every user. If the username is left blank, all local users may transfer files.

b. rermissions

The system can allow access to specific directories, for either users or other systems. The sar ;le format below (in the USERFILE) allows the user **smith** and the system **gemini** to access the directories which follow.

```
smith, gemini  /usr/spool/uucppublic  /usr/spool/news
```

c. Callback

Callback is an effective security measure to ensure that you are connected to an authorized system. Without callback, a remote system may be able to call your local system using the parameters of a valid remote system. The calling system could in fact be an attacker who has acquired the remote system's login and password. Only one system may have the callback option enabled, otherwise a loop will be initiated and no connection will be made.

d. Sample USERFILE

```
, /usr/spool/uucppublic
nuucp, /usr/spool/uucppublic (not needed in BSD 4.2/4.3)
smith, /usr/spool/uucppublic  /ul/smith
root, /
utaurus, taurus  /usr/spool/uucppublic  /usr/src
```

e. Sample of Insecure USERFILES

The following is a sample USERFILE entry which allows all remote systems to transfer files to the public directory and complete access to UUCP for local users.

```
nuucp,/usr/spool/uucppublic
, /
```

The next USERFILE entry allows open access with which all users and all remote systems can transfer files to any directory. [GARF91: pp. 205-206]

```
,/
,/
```

2. L.cmds

The **L.cmds** file (may have other names according to the system used) is a listing of the commands which may be used by remote systems via **uucp**. The **uux** program will not execute any command not found in the **L.cmds** file. For example, if **rmail** is not included, then users will not be able to receive mail from remote systems (via UUCP). Some commands that can cause potential security problems in UUCP are **cat, rm, finger,** and **man**; therefore, commands added to the **L.cmds** file should be selected carefully so that inadvertent security holes are not created. One such security hole is the **ruusend** command, because it allows a remote system to forward files from your system. Table 18 is an example of a **L.cmds** file. [GARF91: p. 208]

TABLE 15: SAMPLE "**L.cmds**" FILE

```
PATH=/bin:/usr/bin:/usr/ucb
rmail
rnews
lpr
who
finger (Use Caution!)
```

D. BNU UUCP (HONEYDANBER UUCP)

In BNU UUCP, the **USERFILE** and **L.cmds** files are replaced with the **Permissions** file. The **Permissions** file allows more control over UUCP security by determining which commands may be executed by the remote system (like the **L.cmds** file) and which files may be accessed. The **Permissions** file is scanned when the **uucico** file is started. There are 13 different permission commands which are allowed by BNU UUCP in the **Permissions** file. The permission commands allow much more flexibility in securing UUCP connections. The permission commands are **MACHINE=, LOGNAME=, REQUEST=, SENDFILES=, PUBDIR=, READ=, WRITE=, NOREAD=, NOWRITE=, CALLBACK=, COMMANDS=, VALIDATE=, & MYNAME=.** Table 16 shows the usage of each of the permission commands. [GARF81: pp. 211-215]

TABLE 16: PERMISSION COMMANDS

COMMAND	MEANING
MACHINE=	Used to denote the specific remote site that is to be contacted by the local system. The system name is inserted after the "=".
LOGNAME=	Used to assign a login name for a remote site which wishes to connect to your system.
REQUEST=	Denotes whether the remote system is allowed to transfer files. The default is "no".
SENDFILES=	Determines whether queued files on the local system will be transferred upon the call of a remote system or wait until the local system initiates the call. The default is "call", which means that files will be transferred when the local calls the remote.
PUBDIR=	Denotes directories specified for public access. Only used in conjunction with LOGNAME entries.

TABLE 16: PERMISSION COMMANDS

COMMAND	MEANING
READ=	Denotes directories which **uucico** can read.
WRITE=	Denotes directories which **uucico** can write.
NOREAD=	Denotes directories which **uucico** can't read. (Overrides the READ command.)
NOWRITE=	Denotes directories which **uucico** can't write. (Overrides the WRITE command.)
CALLBACK=	Denotes whether the local system is required to callback the remote system before conducting a file transfer.
COMMANDS=	Denotes the commands which may be used by remote systems.
VALIDATE=	Specifies the machine name that must be associated with the LOGNAME=.
MYNAME=	Used to change the UUCP name of the local system. This is used for testing but should also be considered a security issue because other systems can change their name and try to log into your system via UUCP.

Note: The **/usr/lib/uucp** directory should be protected by its placement in the NOWRITE list or by not placing it in the WRITE list.

E. UUCP SECURITY HIGHLIGHTS

UUCP control files should be protected, with access limited to the minimum number of directories required. Each remote site should have a unique login whenever possible. Commands which can be executed by remote systems, must be very limited. The final word of caution for UUCP is that if you don't use it, delete it!

XI. NETWORKS

Networks (LAN, WAN, Distributed, etc.) have drastically changed the way in which information is handled. The increased connectivity via computer networks presents many security challenges, especially with operating systems such as Unix. Unix is the predominant operating system used by host computers on the Internet; therefore, Unix security issues can have a significant impact on the network.

A. TCP/IP AND UNIX SECURITY

The Transmission Control Protocol/Internet Protocol provides reliable communications capability for Unix systems connected to the Internet. Each end of the TCP/IP connection has a 16 bit port number. When a connection is made on the Internet, the connection can be defined by the two network addresses (32-bit numbers) and two ports (16-bit numbers). Every network service that is "generally-known" to the Internet has a unique port number which is used to contact the network. An example is port # 513 which is used for the **rlogin** network service.

Port numbers 0 to 1023 are referred to as "trusted ports" in Unix, because only the superuser can originate connections from these ports. If an unauthorized user could use a program to listen to port 513, then it would be possible to receive **rlogin** connections from other users and obtain their passwords.

Other types of systems (non-Unix) can send packets from low numbered ports and Unix will assume the connection is trusted. This means Unix can be spoofed by other systems which do not use the "trusted port" convention.

B. UNIX NETWORK SERVERS

The **/etc/services** and **/etc/inetd.conf** files are used to control network servers. These files determine which servers are run once the system is contacted on a particular communications port.

1. /etc/services

All network services provided by Unix must be listed in the **/etc/services** file. The two primary types of servers are those that run continuously (i.e. **nsfd**) and those that run when required (i.e. **fingerd**). The **/etc/rc** file is used to automatically start up those servers that run continuously.

TABLE 17: FORMAT OF THE "**/etc/services**" FILE

SERVICE NAME,	NETWORK PORT NUMBER/PROTOCOL NAME,	ALIASES
telnet	23/tcp	
smtp	25/tcp	mail
time	39/udp	timeserver

2. /etc/inetd

This program is referred to as the "Internet daemon". It is a server program which listens to the network ports and runs the appropriate server when needed. The program is run by the **/etc/rc** file when the system is booted up. Once started, the **/etc/inetd** file looks at the **/etc/inetd.conf** file, which lists the network services which are to be handled by **/etc/inetd**.

TABLE 18: FORMAT OF THE "**/etc/inetd.conf**" FILE

Components: Service, Socket Type, Protocol Type, Wait/Nowait, User, Command & Argument
telnet stream tcp nowait root /usr/etc/telnetd telnetd
tftp dgram udp wait nobody /usr/etc/tftpd tftpd
echo stream tcp nowait root internal

TABLE 19: FIELDS IN "**inetd.conf**" FILE

NAME	MEANING
Service Name	The service name that is in the /etc/services file.
Socket Type	Specifies whether the service expects to communicate via a datagram or stream.
Protocol Type	TCP or UDP. TCP is used with stream and UDP is used with datagrams.
Wait/Nowait	Wait means the server will expect to process all datagrams received. Nowait means a new server process is executed for each additional connection request received.
User	The UID that the server process is to be run as.
Command & Argument	The command name to be executed and the argument that the command passed.

C. NETWORK SERVICES

1. telnet

Telnet provides a "remote virtual terminal service", with **telnet** as the client program and **telnetd** as the server program. Using **telnet** has the same vulnerability as dialing in via a modem because there is the potential for unauthorized access. An added vulnerability is that if someone has physical access to your network, network analysis (snooping) software can be used to capture packets as they are sent. Since **telnet** is used over the Internet, the potential for a security problem is magnified significantly.

2. rsh & rlogin

Remote terminal service (like **telnet**) is also provided by **rlogin** (client) and **rlogind** (server). The **rlogind** program does not require a username and if coming from a "trusted host", a password is not required. The **rlogin** program can be used over a LAN or the Internet. Use of **rsh** (client) and **rshd** (server) allow the user to run a single command on a remote system and the programs can only work when run from a trusted host. Both **rlogin** and **rsh** can only be used between Berkeley Unix Systems.

3. rexec

Users are allowed to execute commands on remote computers with the **/etc/rexecd** program. It does not require the user to be logged into the remote system; however, the username, password, and the command to be executed are all transmitted over the network and can be vulnerable to snooping.

Error messages are sent back to indicate incorrect usernames or passwords. This feedback could also be utilized by an attacker. To improve security, the **rexec** program can be disabled in the **/etc/inetd.conf** file.

4. finger

The **finger** program is used to find out username, full name, location, telephone number, and login time of users on the system, but the information must be in the **/etc/password** file to be available to **finger**. **Finger** can also be used to find out who is logged onto remote systems. It may be disabled to enhance security, but the trade-off is in limiting contact information to persons trying to conduct business with your organization.

The **/etc/fingerd** program is used to make the finger service available to everyone on the network. The **/etc/fingerd** programs prior to 1988 had a security flaw that was exploited by the Internet Worm in the fall of 1988. All sites should have a newer version of the program installed.

5. FTP (File Transfer Protocol)

FTP is a program which facilitates the transfer of files across a network and between systems. The **ftp** portion of the program is the client and **/etc/ftpd** is the server program. When using **ftp**, the user must login with a username and a password. All **ftp** logins are recorded by the **/usr/adm/wtmp** file on the remote computer. Passwords and usernames are transmitted over the network when using FTP, which causes a security vulnerability when not using anonymous login procedures. Versions of **ftpd** older than 1988 have a security flaw and should be replaced.

a. Anonymous FTP

Anonymous FTP can be used to transfer files from sites which allow this capability. The program **/etc/ftpusers** is a file that lists those users who are not allowed to access files via FTP. All accounts not belonging to specific users should be included in the file. (i.e. root, uucp, news, bin, ingres, daemon, nobody)

b. Security of Anonymous FTP

Incorrectly set up, the anonymous FTP capability can be a serious security vulnerability, giving anyone on the network access to your system. To preclude denial of service, a file quota should be in place so that remote users cannot tie up the system by transferring large files. A common configuration for FTP is for **ftpd** to be set up as its own file system, with **ftp** as the home account and three directories in the account (i.e. **bin, etc, pub**). [GARF91: pp. 245-246]

TABLE 20: COMMANDS TO SET-UP FTP

COMMAND	MEANING
mkdir ~ftp/bin ~ftp/etc ~ftp/pub	Creates FTP directories.
cp /bin/ls ~ftp/bin	Makes a copy of the **ls** program.
chmod 111 ~ftp/bin/ls	Protects the **ls** program.
chmod 111 ~ftp/bin	Protects the **bin** program.
chown root ~ftp/bin	Ensure root owns the directory.
sed -e 's/:[^:]*:/:*:/' /etc/passwd > ~ftp/etc/passwd	Copies the **/etc/passwd** file and changes passwords to asterisks.
sed -e 's/:[^:]*:/:*:/' /etc/group > ~ftp/etc/group	Copies the **/etc/group** file.
chmod 444 ~ftp/etc/*	Makes files in /**etc** not writeable.
chmod 111 ~ftp/etc	Makes the /**etc** directory execute only.
chown root ~ftp/etc	Ensure root owns the directory.
chmod 1777 ~ftp/pub	Makes directory /**pub** writeable by anyone.
chown ftp ~ftp/pub	Ensure that ftp owns the /**pub** directory.
chgrp ftp ~ftp/pub	Changes the group.

TABLE 20: COMMANDS TO SET-UP FTP

COMMAND	MEANING
chmod 555 ~ftp	Make FTP readable and executable by owner, group, & other.
chown root ~ftp	Ensure root owns the **ftp** directory.

Note: Setting the **~ftp/pub** directory to 1777 will allow users on the network to leave files anonymously. This type of access should be approached cautiously.

6. TFTP (Trivial File Transfer Protocol)

This file transfer program is UDP based and does not provide security. Current versions of the program should be restricted so that it can only transfer files to or from a specific directory. This program should not be used unless necessary and then only with the new versions.

7. X Windows (A serious security hazard.)

X Windows is a windowing system for networked computers, which allows a graphical display to be shared by many programs. X-based programs can display their output on any computer on the network. The basic programs in X Windows include **xterm** (terminal emulator), **xclock** (clock display), and **xhost** (a list of hosts from which the X-Window server will accept connections). The **xhost** file will provide at least some security, but any client that connects to the X-server can control the keyboard or mouse, send keystrokes to applications or kill windows.

XII. NIS AND NFS

A. NIS (NETWORK INFORMATION SYSTEM)

NIS is a distributed database that uses a NIS master server to store essential files. The files stored on the NIS master server include host tables, group files, password files, etc. Files on the NIS master server are shared by other computers on the network and appear to the NIS clients as local files.

TABLE 21: FILES NORMALLY MAINTAINED BY NIS

FILE
/etc/aliases
/etc/bootparams
/etc/ethers
/etc/group
/etc/hosts
/etc/hosts.equiv
/etc/netgroup
/etc/networks
/etc/passwd
/etc/protocols
/etc/services

1. System Database Files and NIS

Files such as **/etc/passwd** and **/etc/group** have a special line in the system database file which should begin with a "+" sign. The "+" sign tells any program that scans the client database file to retrieve the rest of the file from the NIS server. The NIS client could have a **/etc/passwd** file format as follows: [GARF91: pp. 256-257]

```
root:sl4nOjf8Q66jJ:0:1:System Administrator:/:/bin/sh
+:*:0:0:::
```

Since the "+" sign refers a program to the NIS server, account information can reside in a single location, allowing easier management of the network. Note: The entry "**+:*:0:0:::**" should not be used in the **/etc/passwd** file on the NIS server. Such an entry would allow anyone to access your system simply by entering a "+" sign at the login prompt.

The line "**+:*:0:0:::**" should be placed in the password file for all NIS clients. This procedure will help system security in the case of the NIS server failing, since there are some implementations of NIS which could allow login with a "+" should the NIS server fail.

2. Netgroups

Netgroups are used to classify users and computers on the NIS network. They are used with NIS or NFS to restrict who may have access to specific services. The database for the netgroup is kept on the NIS master server and resides in the **/etc/netgroup** or **/usr/etc/netgroup** files. [CURR92: p. 90]

TABLE 22: SAMPLE NETGROUP FILE

FORMAT	MEANING
cs4601 (host1,,) (host2,,) (host3,,)	"cs4601" is the Netgroup. The members of the cs4601 netgroup are "(host1,,) (host2,,) (host3,,)".

TABLE 22: SAMPLE NETGROUP FILE

FORMAT	MEANING
Note: Each member of the netgroup is a triple, meaning the structure is in the form (host, username, domain). When the username is left blank, the match comprises all users on the specified host. When the host is left blank, the specified user can be matched with any host in the netgroup. The domain is normally left blank.	

B. NFS (NETWORK FILESYSTEM)

NFS allows sharing of files by many computers, over a network. The NFS server is normally a drive with mass storage capability, which can have files that are accessed by many client computers. The clients mount the disk drive of the server, making it appear as if it were a local disk drive. Many local computers can share the server at the same time, as well as remote computers.

NFS server programs written for other operating systems give the capability of accessing files across different platforms. By using NFS, files on the server can be read or modified without logging into the server or supplying a password. Most NFS security problems arise from its inherent capability.

1. Mount and NFS

The mount and NFS protocols are used to determine which filesystems are available to which hosts. When a file system is mounted, NFS can execute the following functions: CREATE, MKDIR, REMOVE, RENAME, RMDIR, LINK, LOOKUP, ADDR, and SYMLINK.

The combination of mount and NFS are powerful, but the underlying security implications must be considered by the system administrator. An example command follows:

/etc/mount gemini:/usr /localusr (This command mounts the directory **/usr** on a server named **gemini** in a local directory called **/localusr**.)

2. /etc/exports

The **/etc/exports** file determines what kind of access a client may have on a mounted filesystem, as well as which clients are allowed to mount the server filesystem. Sample file entry: [GARF91: p. 263]

```
/ -access=csdept, root=admin
/usr -ro
/usr/spool/mail -access=csdept
```

The preceding commands allow anyone in the "csdept" group to mount the root directory; however, only the group "admin" has root privilege. The directory **/usr** is exported as a Read-Only filesystem and is available to anyone on the Internet. The directory **/usr/spool/mail** is exported so that anyone in the group "csdept" has access. Table 26 gives the various security options available. [GARF91: pp. 263-264]

** *filesystems should not be exported unless absolutely necessary* **

TABLE 23: SECURITY OPTIONS FOR "**/etc/exports**"

OPTION	MEANING
access=machinelist	Exports the directory as Read-Only to those hosts or netgroups which are listed.
rw=machinelist	Exports the filesystem as Read-Write to those hosts or netgroups listed, and Read-Only to other hosts.
root=machinelist	This entry allows the superuser on the listed hosts to have superuser access on the server.

TABLE 23: SECURITY OPTIONS FOR "/etc/exports"

OPTION	MEANING
anon=uid	This entry determines what UID will be used for those NFS requests that do not have a UID. Access can be disallowed by entering a -1.
secure	Directs NFS to use the Sun AUTH_DES authentication system.

Note: The security options pertain to the filesystem as a whole and not to individual directories.

3. /usr/etc/showmount

The **/usr/etc/showmount** command can be used by the NFS server to determine which clients have mounted directories from the server. The command can also show which filesystems are exported, which directories have been mounted remotely, and lists all hosts to include the directories which are mounted.

C. METHODS TO IMPROVE NFS SECURITY

Exported filesystems should be restricted to only those that are essential. If a filesystem must be exported, it should be exported as Read-Only when feasible. Machines that are exported to should also be restricted.

The NFS server should be owned by root. Executable programs used by NFS should not be exported. If the programs must be exported, ensure they are Read-Only. Filesystems should be exported as Read-Write only to "trusted" computers. Finally, world-writable directories should never be exported. (i.e. **/tmp, /usr/tmp, and /usr/spool/uucppublic**)

D. METHODS TO IMPROVE NIS SECURITY

NIS is basically insecure. On most implementations of NIS, an attacker can get a copy of the databases exported by the NIS server. The database can disclose the distributed password file. Security can be improved if the **ypserv** program is modified so that it will only respond to authorized hosts on the network.

The **ypbind** program should be used with the -**secure** flag, so that it will not accept data from a false **ypserv** server. It is important to remember that most, if not all, security precautions can be rendered useless, if the attacker gains root access on the local network.

E. THE BOTTOM LINE

<u>If security is a major concern on the network in question, do not use NFS or NIS</u>.

REFERENCES

[ANDE86] Anderson, Gail and Anderson, Paul. The UNIX C Shell Field Guide. Englewood Cliffs, NJ: Prentice-Hall, 1986.

[BALD87] Baldwin, Robert W. Rule Based Analysis of Computer Security. Massachusetts Institute of Technology, June 1987.

[BELS93] Belsie, Laurent, The Electronic Village. World Monitor, Vol. 6, No. 3, March 1993.

[CONG89] One Hundred First Congress. "Bugs in the Program: Problems in Federal Government Software Development and Regulation". Staff Study by the Subcommittee on Investigations and Oversight, U. S. House of Representatives, July 1989.

[CONG91] One Hundred Second Congress. "Computer Security". Hearing before the Subcommittee on Technology and Competitiveness of the Committee on Science, Space, and Technology, U. S. House of Representatives, June 1991.

[COPS91] Farmer, Dan. COPS Report. A text file downloaded from the Internet. January 1991.

[CUNN90] Cunningham, William C. and Strauchs, John J. and Van Meter, Clifford W. "Private Security Trends 1970-2000." The Hallcrest Report II. Stoneham, MA: Butterworth-Heinemann,1990.

[CURR92] Curry, David A. "A Guide for Users and System Administrators." UNIX System Security. Reading, MA: Addison-Wesley Publishing, 1992.

[DCAC] DCAC 310-P115-1. Communications Security: DDN Security Management Procedures for Host Administrators. DCA Circular, Volume I.

[DDN91] DDN NIC, DDN New User Guide, Chantilly, VA: 3rd Edition, 1991.

[DOD85] DOD 5200.28-STD. Department of Defense Trusted Computer System Evaluation Criteria. Washington, D.C.: Assistant Secretary of Defense, Command, Control, Communications, and Intelligence, 1985.

[DOD85A] CSC-STD-002-85. Department of Defense Password Management Guideline. Washington, D.C.: DOD Computer Security Center, 1985.

[FERB93] Ferbrache, David and Shearer, Gavin. Unix Installation Security & Integrity. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[GARF91] Garfinkel, Simson and Spafford, Gene. Practical UNIX Security. Sebastopol, CA: O'Reilly & Associates, 1991.

[HEDG83] Hedges, Robert L. Computer Science Training in the Department of Defense: The Silent Problem. Fort Lesley J. McNair, Washington, D.C. National Defense University Press, 1983.

[HOLB91] Holbrook, P. and Reynolds, J., Editors, Site Security Handbook. RFC 1244, Site Security Policy Handbook Working Group, July 1991.

[KEHO92] Kehoe, Brendan P. Zen and the Art of the Internet, 1992.

[KROL87] Krol, Ed, The Hitchhikers Guide to the Internet. University of Illinois, August 1987.

[KROL93] Krol, Ed, The Whole Internet: User's Guide & Catalog, Sebastopol, CA: O'Reilly & Associates, 1993.

[MADR92] Madron, Thomas W. Network Security in the 90's: Issues and Solutions for Managers. New York, NY: John Wiley & Sons, 1992.

[MUFF92] Muffet, Alec D. E. "Crack Version 4.1", A Sensible Password Checker for Unix. March 3, 1992.

[NCSC87] NCSC-TG-005, Version-1. Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria. Ft. George G. Meade, MD: National Computer Security Center, 1987.

[NSDD84] NSDD 145. National Policy on Telecommunications and Automated Information Systems Security. National Security Decision Directive. September, 1984.

[NTIS87] National Telecommunications and Information Systems Security (NTISS). Advisory Memorandum on Office Automation Security Guideline, NTISSAM COMPUSEC/1-87, 16 January 1987.

[PFLE89] Pfleeger, Charles P. Security in Computing. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[QUAR90] Quarterman, John. S., The Matrix: Computer Networks and Conferencing Systems Worldwide, Digital Equipment Corporation, 1990.

[REYN89] Reynolds, J. The Helminthiasis of the Internet. RFC 1135, Network Working Group, December 1989.

[RICH92] Rich, Lyford D. "Unix Security: A Penetration Analysis of Navy Computer Systems." Masters Degree Thesis. Monterey, CA: Naval Postgraduate School, 1992. (Limited Distribution)

[RUSS91] Russell, Deborah and Gangemi, G.T. Sr. Computer Security Basics. Sebastopol, CA: O'Reilly & Associates, 1991.

[SOBE89] Sobell, Mark G. A Practical Guide to the UNIX System. Redwood City, CA: Benjamin/Cummins Publishing, 1989.

[WOOD85] Wood, Patrick H. and Kochan, Stephen G. UNIX System Security. Carmel, IN: Hayden Books, 1985.

[WOOD87] Wood, Charles C. et. al. Computer Security: A Comprehensive Controls Checklist, New York, NY: John Wiley & Sons, 1987.

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School 555 Dyer Rd. Room 229 Monterey, CA 93943-5002	2
3.	Chairman Administrative Science Department Code AS/Wp Naval Postgraduate School 555 Dyer Rd. Room 229 Monterey, CA 93943-5103	1
4.	Dr. Carl Jones Professor, Administrative Science Department Code AS/Js 555 Dyer Rd. Room 229 Monterey, CA 93943-5103	1
5.	Mr. Roger Stemp Visiting Instructor, Computer Science Department Code CSSp Naval Postgraduate School Monterey, CA 93943-5000	1
6.	LT Thomas L. Brown c/o 4088 Beech Bluff Rd. Beech Bluff, TN 38313	1